

162-TD-001-004

Science Software I&T Operational Procedures for the ECS Project (aka, The Green Book)

Technical Data

February 1998

Prepared Under Contract NAS5-60000

RESPONSIBLE ENGINEER

Dominick P. Iascone, Jr. /s/ 2/20/98

Dominick P. Iascone, Jr. Date
EOSDIS Core System Project

SUBMITTED BY

David E. Manion /s/ 2/20/98

David Manion, SSI&T Manager Date
EOSDIS Core System Project

Raytheon Systems Company
Upper Marlboro, Maryland

This page intentionally left blank.

Abstract

The purpose of this Technical Data Paper (also referred to as the “Green Book”) is to delineate the operational procedures to accomplish the various steps that may be involved in the integration and test of Science Data Production Software (SDPS/W) with the Earth Observing System (EOS) Data and Information System (EOSDIS) Core System (ECS). The SDPS/W integration and test (SSI&T) is performed at the Distributed Active Archive Center (DAAC) responsible for the generation of the standard products.

General information concerning preparing and delivering SDPS/W to the DAAC is found in the *Science User’s Guide and Operations Procedure Handbook for the ECS Project, Part 4: Software Developer’s Guide to Preparation, Delivery, Integration and Test with ECS* (205-CD-002-006). Each DAAC and Instrument Team (IT) combination have formulated specific agreements, understandings, or procedures that will guide their respective SSI&T activities. The procedures in this document provide detailed instructions on how to use the tools that are provided in the Release B of ECS to accomplish the steps outlined in the DAAC-IT procedures.

This page intentionally left blank.

Table of Contents

Abstract

1. Introduction

1.1 Purpose.....	1-1
1.2 Organization	1-1
1.3 Changes from the Previous Version.....	1-2
1.4 Review and Approval.....	1-2

2. Related Documentation

2.1 Parent Documents	2-1
2.2 Applicable Documents	2-1
2.3 Information Documents	2-2

3. General Information

3.1 SSI&T Road Map	3-1
3.2 Set Up of SSI&T User Accounts	3-4
3.3 How to Use The Procedures.....	3-4

4. Pre-Release B Testbed and B.0 Architectural Overview: Impact on SSI&T Procedures

4.1 Pre-Release B Testbed Architecture: Overview	4-1
4.2 Release B.0 Architecture: Overview	4-2
4.2.1 Client Subsystem (CLS).....	4-4
4.2.2 Interoperability Subsystem (IOS).....	4-4
4.2.3 Data Management Subsystem (DMS)	4-4
4.2.4 Data Server Subsystem (DSS)	4-4

4.2.5 Ingest Subsystem (INS).....	4-5
4.2.6 Data Processing Subsystem (DPS).....	4-5
4.2.7 Planning Subsystem (PLS)	4-5
4.2.8 Communications Subsystem (CSS).....	4-6
4.2.9 Management Subsystem (MSS)	4-6
4.2.10 Internetworking Subsystem (ISS).....	4-7
4.3 Implications for SSI&T Procedures.....	4-8

5. Configuration Management

5.1 Creating a View in ClearCase.....	5-1
5.2 Setting a View in ClearCase	5-2
5.3 Importing Multiple Files into ClearCase from a Directory Structure.....	5-3
5.4 Entering a Single File into ClearCase	5-4
5.5 Entering a New Directory into ClearCase.....	5-6
5.6 Checking Out an Element From ClearCase.....	5-7
5.7 Checking a Modified Element into ClearCase.....	5-8

6. The ECS Assistant

6.1 Using ECS Assistant to Start Up / Shut Down Servers.....	6-1
6.2 Using ECS to Perform System Monitoring.....	6-6
6.2.1 Using ECS Assistant to Open / View Log Files for a Selected Server.....	6-6
6.2.2 Using ECS Assistant to Monitor Server Status.....	6-7
6.3 Using ECS Assistant to Manage ESDTs.....	6-11
6.3.1 Adding ESDTs to the Storage Area	6-11
6.3.2 Removing ESDTs via the Science Data Server.....	6-14
6.3.3 Using ECS Assistant to View ECS Science Data Server Database	6-16

7. The SSIT Manager

7.1 General Set Up and Running of the SSIT Manager.....	7-3
7.2 Set Up of a Checklist for the SSIT Manager	7-4

8. DAP Insert

8.1 Performing a DAP Insert	8-1
-----------------------------------	-----

9. DAP Acquire

9.1 Performing a DAP Acquire Using SSIT Manager.....	9-1
--	-----

10. PGE Checkout

10.1 Checking for ESDIS Standards Compliance in FORTRAN 77.....	10-1
10.2 Checking for ESDIS Standards Compliance in Fortran 90.....	10-4
10.3 Checking for ESDIS Standards Compliance in C.....	10-6
10.4 Checking for ESDIS Standards Compliance in Ada.....	10-8
10.4.1 Checking for ESDIS Standards Compliance in Ada: Verdex COTS.....	10-9
10.4.2 Checking for ESDIS Standards Compliance in Ada: GNU <i>gcc</i> Compiler.....	10-11
10.5 Checking for Prohibited Functions.....	10-12
10.5.1 Checking for Prohibited Functions: GUI Version.....	10-12
10.5.2 Checking for Prohibited Functions: Command-Line Version.....	10-16
10.6 Checking Process Control Files.....	10-18
10.6.1 Checking Process Control Files: GUI Version.....	10-18
10.6.2 Checking Process Control Files: Command-Line Version.....	10-21
10.7 Extracting Prologs.....	10-22

11. Compiling and Linking Science Software

11.1 Updating the Process Control Files (PCFs).....	11-1
11.2 Setting Up the SDP Toolkit Environment.....	11-4
11.3 Compiling Status Message Facility (SMF) Files.....	11-7
11.4 Building Science Software with the SCF Version of the SDP Toolkit.....	11-9
11.5 Building Science Software with the DAAC Version of the SDP Toolkit.....	11-12

12. Running a PGE in a Simulated SCF Environment

12.1 Setting Up the Environment for Running the PGE.....	12-1
12.2 Running and Profiling the PGE.....	12-3

13. PGE Registration and Test Data Preparation

13.1 PGE Registration.....	13-1
13.1.1 Updating the PDPS Database with ESDT Metadata	13-1
13.1.2 Updating the PDPS Database with PGE Metadata.....	13-3
13.1.3 Updating the PDPS Database with Operational Metadata	13-7
13.2 Test Data Preparation.....	13-9
13.2.1 Generating a Metadata Configuration File (Source MCF).....	13-10
13.2.2 Creating a Target MCF (.met) for a Dynamic/Static Granule	13-11
13.2.3 Inserting Dynamic Data Granules to the Data Server	13-25
13.2.4 Inserting Static Data Granules to the Data Server	13-27

14. PGE Planning, Processing, and Product Retrieval

14.1 Using the Production Request Editor	14-1
14.1.1 Invoking the Production Request Editor.....	14-1
14.1.2 Defining a New Production Request	14-4
14.1.3 Viewing Production Requests	14-7
14.1.4 Viewing Data Processing Requests	14-10
14.1.5 Listing Data Processing Requests.....	14-10
14.2 Using the Production Planning Workbench	14-11
14.2.1 Using the Planning Workbench to Run One PGE	14-11
14.3 Monitoring Production	14-14
14.4 Using the Q/A Monitor	14-16

15. File Comparison

15.1 Using the GUI HDF File Comparison Tool.....	15-1
15.2 Using the hdiff HDF File Comparison Tool	15-2
15.3 Using the ASCII File Comparison Tool.....	15-3
15.4 Using the Binary File Difference Assistant	15-4

16. Data Visualization

16.1 Viewing Product Metadata with the EOSView Tool.....	16-1
16.2 Viewing Product Data with the EOSView Tool.....	16-3
16.2.1 Viewing HDF Image Objects.....	16-5
16.2.2 Viewing HDF-EOS Grid Objects.....	16-7
16.2.3 Viewing HDF-EOS Swath Objects	16-10
16.2.4 Viewing HDF SDS Objects	16-13
16.3 Viewing Product Data with the IDL Tool.....	16-15
16.3.1 Creating an Image Display Using IDL.....	16-16
16.3.2 Saving an Image Display Using IDL	16-20
16.3.3 Creating a Plot Display Using IDL.....	16-22
16.3.4 Saving a Plot Display Using IDL.....	16-23
16.3.5 Raster Math Fundamentals Using IDL.....	16-23
16.4 Raster Mapping Fundamentals.....	16-24

17. Placing the Science Software Executable on the Data Server

17.1 Assembling a Science Software Executable Package.....	17-1
17.2 Inserting a Science Software Executable Package to the Data Server	17-7

18. Inserting a Science Software Archive Package

18.1 Inserting an SSAP into PDPS	18-2
--	------

19. Inserting a Static File from Outside the DAP

19.1 Inserting a DAP ancillary file into PDPS	19-1
---	------

20. Updating a Science Software Archive Package

20.1 Updating an SSAP	20-1
-----------------------------	------

21. Post-SSI&T Activities

21.1 Validating Inventory Metadata Updates.....	21-1
21.2 Reporting Problems.....	21-1
21.2.1 Setting Up the Environment for DDTs.....	21-1
21.2.2 Submitting a New Problem Report	21-2
21.2.3 Changing or Revising a Problem Report.....	21-5

22. Troubleshooting and General Investigation

22.1 Examining PGE Log Files	22-1
22.1.1 Log Files From PGEs Run Outside of the PDPS	22-1
22.1.2 Production History Log Files From PGEs Run Within the PDPS	22-3
22.1.3 History Log Files From Failed PGEs Run Within the PDPS	22-5
22.2 The Production History	22-6
22.3 Examining PDPS-Related Scripts and Message Files.....	22-8
22.3.1 Examining AutoSys JIL Scripts.....	22-9
22.3.2 Examining Application Log Files.....	22-9

23. Miscellaneous

23.1 Setting Up Environment for Direct PDPS Database Access.....	23-1
--	------

Figures

4.1 B.0 Context Diagram.....	4-3
6.1-1 ECS Assistant GUI.....	6-3
6.1-2 Subsystem Manager GUI	6-4
6.2.2-1 Server Monitor GUI.....	6-9
6.2.2-2 cdsping GUI.....	6-10
6.3.1-1 ESDT Manager GUI.....	6-12
6.3.3-1 Database Login GUI	6-17
6.3.3-2 DB Viewer GUI.....	6-18
9.1-1 Selecting DAP Acquire from SSIT Manager.....	9-3

10.1-1 Screen Snapshot: Accessing FORCHECK from SSIT Manager	10-3
10.5.1-1 Invoking the Prohibited Function Checker	10-15
10.5.1-2 Starting Screen for Prohibited Function Checker	10-16
10.5.1-3 File Selection Menu	10-16
10.5.1-4 Selected Files	10-16
10.5.1-5 Results Screen.....	10-16
10.6.1-1 Invoking the Process Control File Checker.....	10-20
10.6.1-2 Selecting a File to Check	10-20
14.1.1-1 Production Request Editor (Planning) GUI.	14-3
14.1.2-1 PR Editor GUI.....	14-6
14.1.3-1 PR List GUI.....	14-9
14.2.1-1 Planning Workbench GUI.	14-13

Tables

3.1-1 Road Map of SSI&T for a PGE	3-1
4.1 Major SSI&T Procedural Differences: Testbed to B.0 Releases	4-8
5.1-1 Creating a View in ClearCase - Quick-Step Procedures.....	5-2
5.2-1 Setting a View in ClearCase - Quick-Step Procedures.....	5-3
5.3-1 Importing Multiple Files into ClearCase from a Directory Structure - Quick-Step Procedures.....	5-4
5.4-1 Entering a Single File into ClearCase - Quick-Step Procedures.....	5-6
5.5-1 Entering a New Directory into ClearCase - Quick-Step Procedures	5-7
5.6-1 Checking Out an Element From ClearCase - Quick-Step Procedures	5-8
5.7-1 Checking a Modified Element into ClearCase - Quick-Step Procedures.....	5-9
6.1-1 Server Start Up/Shut Down Using ECS Assistant - Quick-Step Procedures	6-5
6.2.1-1 Using ECS to View the Logfile - Quick-Step Procedures	6-7
6.2.2-1 Using ECS Monitor GUI - Quick-Step Procedures.....	6-11
6.3.1-1 Running ESDT Add GUI - Quick-Step Procedures	6-13
6.3.2-1 Using ECS Assistant to Remove ESDTs - Quick-Step Procedures.....	6-15

6.3.3-1 Using ECS Assistant to View ECS Science Data Server Database - Quick-Step Procedures	6-19
7.1-1 Assumptions and Limitations	7-3
7.1-2 General Set Up and Running of the SSIT Manager - Quick-Step Procedures	7-4
7.2-1 Assumptions and Limitations	7-5
7.2-2 Set Up of a Checklist for the SSIT Manager - Quick-Step Procedures	7-7
8.1-2 Performing a DAP Insert - Quick-Step Procedures	8-3
9.1-1 Performing a DAP Acquire - Quick-Step Procedures	9-3
10.1-1 Checking for ESDIS Standards Compliance in FORTRAN 77 - Quick-Step Procedures	10-4
10.2-1 Checking for ESDIS Standards Compliance in Fortran 90 - Quick-Step Procedures	10-6
10.3-1 Checking for ESDIS Standards Compliance in C - Quick-Step Procedures	10-8
10.4.1-1 Checking for ESDIS Standards Compliance in Ada: Verdex COTS - Quick-Step Procedures	10-10
10.4.2-1 Checking for ESDIS tandards Compliance in Ada: GNU gcc Compiler - Quick-Step Procedures	10-12
10.5.1-1 File Name Extensions Recognized	10-13
10.5.1-2 Checking for Prohibited Functions: GUI Version - Quick-Step Procedures	10-15
10.5.2-1 Checking for Prohibited Functions: Command-Line Version - Quick-Step Procedures	10-18
10.6.1-1 Checking Process Control Files: GUI Version - Quick-Step Procedures	10-19
10.6.2-1 Checking Process Control Files: Command-Line Version - Quick-Step Procedures	10-22
10.7-1 Extracting Prologs - Prolog Delimiters	10-23
10.7-2 Extracting Prologs - File Name Extensions	10-23
10.7-3 Extracting Prologs - Quick-Step Procedures	10-24
11.1-1 Updating the Process Control Files (PCFs) - Quick-Step Procedures	11-3
11.2-1 Object Types on the SGI	11-4
11.2-2 SDP Toolkit Versions at the DAAC	11-5
11.2-3 Setting Up the SDP Toolkit Environment - Quick-Step Procedures	11-7
11.3-1 Compiling Status Message Facility (SMF) Files - Quick-Step Procedures	11-9

11.4-1 Building Science Software with the SCF Version of the SDP Toolkit - Quick-Step Procedures	11-12
11.5-1 Building Science Software with the DAAC Version of the SDP Toolkit - Quick-Step Procedures	11-15
12.1-1 Setting Up the Environment for Running the PGE - Quick-Step Procedures.....	12-2
12.2-1 Running and Profiling the PGE - Quick-Step Procedures	12-4
13.1.1-1 Updating the PDPS Database with ESDT Metadata - Quick-Step Procedures.....	13-3
13.1.2-1 Updating the PDPS Database with PGE Metadata - Quick-Step Procedures	13-6
13.1.3-1 Updating the PDPS Database with Operational Metadata - Quick-Step Procedures	13-9
13.2-1 Test Data Preparation - Activity Checklist.....	13-10
13.2.1-2 Generating a Source MCF for an ESDT - Quick-Step Procedures	13-11
13.2.3-1 Inserting Dynamic Data Granules to the Data Server - Quick-Step Procedures	13-26
13.2.4-1 Inserting Static Data Granules to the Data Server - Quick-Step Procedures.....	13-28
14.1.1-1 Invoking the Production Request Editor - Quick-Step Procedures	14-4
14.1.2-1 Defining a New Production Request - Quick-Step Procedures	14-7
14.1.3-1 Viewing Production Requests - Quick-Step Procedures	14-10
14.1.4-1 Viewing Data Processing Requests - Quick-Step Procedures.....	14-10
14.1.5-1 Listing Data Processing Requests - Quick-Step Procedures	14-11
14.2.1-1 Using the Planning Workbench to Run One PGE - Quick-Step Procedures	14-12
14.3-1 AutoSys Jobs for a DPR.....	14-14
14.3-2 Monitoring Production - Quick-Step Procedures	14-16
14.4-1 Using the Q/A Monitor - Quick-Step Procedures	14-18
15.1-1 Assumptions and Limitations	15-1
15.1-2 Using the GUI HDF File Comparison Tool - Quick-Step Procedures	15-2
15.2-1 Assumptions and Limitations	15-2
15.2-2 Using the hdiff HDF File Comparison Tool - Quick-Step Procedures	15-3
15.3-1 Assumptions and Limitations	15-3
15.3-2 Using the ASCII File Comparison Tool - Quick-Step Procedures	15-4
15.4-1 Assumptions and Limitations	15-4

15.4-2 Using the Binary File Difference Assistant - Quick-Step Procedures	15-6
16.1-1 Viewing Product Metadata with the EOSView Tool - Quick-Step Procedures	16-3
16.2-1 Viewing Product Data with the EOSView Tool - Quick-Step Procedures	16-5
16.2.1-1 Viewing HDF Image Objects - Quick-Step Procedures	16-7
16.2.2-1 Viewing HDF-EOS Grid Objects - Quick-Step Procedures.....	16-10
16.2.3-1 Viewing HDF-EOS Swath Objects - Quick-Step Procedures	16-13
16.2.4-1 Viewing HDF SDS Objects - Quick-Step Procedures.....	16-15
16.3-1 Viewing Product Data with the IDL Tool - Quick-Step Procedures	16-16
16.3.1-1 Creating an Image Display Using IDL - Activity Checklist.....	16-17
16.3.1-2 Creating an Image Display Using IDL - Quick-Step Procedures.....	16-19
16.3.2-1 Saving an Image Display Using IDL - Activity Checklist	16-20
16.3.2-2 Saving an Image Display Using IDL - Quick-Step Procedures.....	16-22
16.4-1 Raster Mapping Fundamentals - Activity Checklist.....	16-24
16.4-2 Raster Mapping Fundamentals - Quick-Step Procedures.....	16-27
17.1-1 Assembling a Science Software Executable Package- Quick-Step Procedures.....	17-3
17.2-1 Inserting a Science Software Executable Package to the Data Server - Quick-Step Procedures.....	17-8
21.2.1-1 Setting Up the Environment for DDTs - Quick-Step Procedures	21-2
21.2.2-1 Submitting a New Problem Report - Quick-Step Procedures.....	21-4
22.1.1-1 Log Files From PGEs Run Outside of the PDPS - Quick-Step Procedures.....	22-3
22.1.2-1 Production History Log Files From PGEs Run Within the PDPS - Quick-Step Procedures.....	22-5
22.1.3-1 History Log File From Failed PGEs Run Within the PDPS - Quick-Step Procedures.....	22-6
22.2-1 The Production History - Quick-Step Procedures.....	22-8
22.3.1-1 Examining AutoSys JIL Scripts - Quick-Step Procedures	22-9
23.1-1 Setting Up Environment for Direct PDPS Database Access - Quick-Step Procedures.....	23-2
23.1-2 Examining Application Log Files - Quick-Step Procedures	23-3

1. Introduction

1.1 Purpose

The purpose of this Technical Data Paper (also referred to as the “Green Book”) is to delineate the operational procedures to accomplish the various steps that may be involved in the integration and test of Science Data Production Software (SDPS/W) with the Earth Observing System (EOS) Data and Information System (EOSDIS) Core System (ECS). The SDPS/W integration and test (SSI&T) is performed at the Distributed Active Archive Center (DAAC) responsible for the generation of the standard products.

General information concerning preparing and delivering SDPS/W to the DAAC is found in the *Science User’s Guide and Operations Procedure Handbook for the ECS Project, Part 4: Software Developer’s Guide to Preparation, Delivery, Integration and Test with ECS* (205-CD-002-006). Each DAAC and Instrument Team (IT) combination have formulated specific agreements, understandings, or procedures that will guide their respective SSI&T activities. The procedures in this document provide detailed instructions on how to use the tools that are provided in the Release B of ECS to accomplish the steps outlined in the DAAC-IT procedures.

1.2 Organization

Section 2 lists related documentation. Section 3 is devoted to general information concerning SSI&T including a road map for integrating a PGE into the ECS. Section 4 gives an overview of the Pre-Release B Testbed and B.0 architectures. Section 5 contains procedures related to configuration management. Section 6 covers the usage of the ECS Assistant GUI. Section 7 describes set up, preparation, and use of the SSIT Manager, the main SSI&T tool provided at the DAACs. Section 8 gives instructions for performing a DAP insert. Section 9 gives instructions for performing a DAP acquire. Section 10 gives the procedures dealing with standards checking and compliance of the science software. Section 11 describes how to build the delivered science software and Section 12 describes how to run it in a simulated SCF environment.

Section 13 describes the procedures for integration of the science software into the ECS; it describes procedures for registering the PGE with the Planning and Data Processing System (PDPS) and for inserting data files to the Science Data Server. Section 14 describes procedures for planning and running science software within the PDPS and retrieving the output. Section 15 then describes procedures for comparing the output produced to that delivered.

Section 16 contains procedures for examining output in more detail using some of the data visualization tools provided.

Section 17 covers the assembly of an SSEP and placing it onto the Data Server. Section 18 covers the procedures for inserting an SSAP into PDPS and Section 19 deals with inserting a static file from outside the DAP into PDPS. Section 20 covers the procedures for updating an SSAP.

Section 21 describes procedures for activities that are post-SSI&T including problem reporting. Section 22 describes some procedures for investigating problems that occur during SSI&T either with the science software or with the ECS software.

Section 23 contains miscellaneous procedures that do not fit into the previous sections.

Appendix A describes the Science Data Server in its handling of metadata. Appendix B contains a sample checklist file used with the SSIT Manager. Appendix C contains a table of environment variables that are needed and used by various SSI&T tools. Appendix D contains sample Target MCFs for a variety of files that are to be Inserted to the Science Data Server. Appendix E contains sample ODL files that are used in the PGE registration process. Finally, Appendix F contains quick step procedures for registering a PGE.

1.3 Changes from the Previous Version

The tools for handling ESDTs have been modified and the corresponding sections, 5.1 and 5.2, have been changed to show this. Section 12.6 has been added with instructions on using the Reaper which may be needed in some situations. A description on accessing the history logs of failed PGE runs is included in Section 16. Appendix F, also has been added that includes two specific scenarios for registering a PGE and for clearing the PDPS database. The attribute for the MCFs, “RangeBeginningDateTime”, should not be used and the sample MCFs reflect this in Section 11. Finally minor typographical errors have been corrected.

1.4 Review and Approval

This Technical Data Paper is an informal document approved at the Office Manager level. It does not require formal Government review or approval; however, it is submitted with the intent that review and comments will be forthcoming.

Questions regarding technical information contained within this paper should be addressed to the following ECS contact:

Karin Loya, SSI&T Manager
(301) 925-1052
kloya@eos.hitc.com

2. Related Documentation

The following related documentation is available either in hard copy from the author or generating source or on the World Wide Web (WWW) at the URL listed. Additionally, most documents generated by the ECS project are available on the WWW via the ECS Data Handling System at <http://edhs1.gsfc.nasa.gov/>.

2.1 Parent Documents

The parent documents are the documents from which this document's scope and content are derived.

162-TD-001-003	Science Software I&T Operational Procedures for the ECS Project (aka, the Green Book)
609-CD-001-001	Interim Release One (Ir1) Maintenance and Operations Procedures for the ECS Project
609-DR-002-001	Release A Operations Tools Manual for the ECS Project

2.2 Applicable Documents

The following documents are referenced within and are directly applicable to this document:

423-16-01	Data Production Software and Science Computing Facility (SCF) Standards and Guidelines, Revision A, October 1996
205-CD-002-003	Science User's Guide and Operations Procedure Handbook for the ECS Project, Volume 4: Software Developer's Guide to Preparation, Delivery, Integration and Test with ECS, Revision 1
333-CD-003-005	Release A SCF Toolkit Users Guide for the ECS Project
445-TP-006-001	EOSView Users Guide for the ECS Project
	FORCHECK for Sun/SunOS, A Fortran Verifier and Programming Aid, Installation Guide
	FORCHECK for Sun/SunOS, A Fortran Verifier and Programming Aid, Users Guide
	IDL Basics, v3.5, November 1993, Research Systems, Inc., Boulder, CO

IDL Reference Guide, v4, Volume 1, Routines A-M, March 1995, Research Systems, Inc., Boulder, CO

IDL Reference Guide, v4, Volume 2, Routines N-Z, March 1995, Research Systems, Inc., Boulder, CO

IDL User's Guide, v4, March 1995, Research Systems, Inc., Boulder, CO

PureDDTS (Distributed Defect Tracking System) 3.2.1 User's Manual, Pure Software (support@pure.com), Part Number PDT320-XPX-UGD

2.3 Information Documents

The following documents, although not referenced herein or directly applicable, do amplify or clarify the information presented in this document:

X3.9-1978	ANSI FORTRAN 77 Programming Language Standard
305-CD-010-001	Release A SDPS Planning Subsystem Design Specification for the ECS Project
416-WP-001-001	Implementation Plan for the Pre-Release B Testbed for the ECS Project

Please note that Internet links cannot be guaranteed for accuracy or currency.

World Wide Web pages...

Prohibited Functions in the EOSDIS Core System at URL:
<http://ecsinfo.hitc.com/iteams/ProhibFunc/>

FORTRAN 77 in the EOSDIS Core System at URL:
<http://ecsinfo.hitc.com/iteams/Standards/F77std.html>

PGE Design Information Page at URL:
http://ecsinfo.hitc.com/iteams/Science/pge_wp.html

EOS Instrument Team Information Page at URL:
<http://ecsinfo.hitc.com/iteams/>

3. General Information

3.1 SSI&T Road Map

Science software integration and test (SSI&T) is the process by which science software is tested for production readiness in the DAACs. The goals are to assure (1) *reliability*, which means that the software runs to normal completion repeatedly over the normal range of data inputs and runtime conditions, and (2) *safety*, which means that the software executes without interfering with other software or operations.

SSI&T activities can be broadly separated into two categories: pre-SSI&T and formal SSI&T. Pre-SSI&T activities can be defined as those activities that do not involve the ECS Planning and Data Processing System (PDPS) or the Science Data Server. Formal SSI&T activities are defined as those that involve the full ECS including the PDPS and Science Data Server.

Table 3.1-1 can be used as a PGE road map for the remainder of the Green. It lists most SSI&T activities in the order in which they are most likely to occur. Column 1 describes the SSI&T task; column 2 lists the relevant sections in the Green Book; and column 3 represents an attempt to qualify the importance of the task. Tasks labeled as “Critical” must be done in order to have the PGE integrated into the ECS and run by the automated PDPS. Tasks labeled as “Essential”, “Valuable”, and “Optional” have lowering degrees of importance, respectively, and are somewhat subjective. This table should not be misconstrued as overriding specific agreements between DAACs and Instrument Teams.

Table 3.1-1. Road Map of SSI&T for a PGE

Step	Task	Sections	Importance
1	Acquire and unpack the Delivered Algorithm Package (DAP).	9	Critical
2	Inspect the delivery and accompanying documentation.	N/A	Valuable
3	Place the DAP contents under configuration management.	5	Valuable
4	Check the source files for standards compliance.	10.1,10.2, 10.3, 10.4	Valuable
5	Check the source files for prohibited functions.	10.5	Valuable
6	Extract and check prologs.	10.7	Valuable
7	Check the Process Control File (PCF).	10.6	Valuable

Table 3.1-1. Road Map of SSI&T for a PGE (cont.)

Step	Task	Sections	Importance
8	Revise the delivered PCF to comply with the local DAAC environment and the location of data files.	11.1	Essential
9	Compile the SDP Toolkit Status Message Facility (SMF) files.	11.3	Essential
10	Build the PGE linking it to the SCF version of the SDP Toolkit.	11.4	Essential
11	Run and profile the PGE from the command line. Save the profiling results. They will be used later when entering operational metadata into the PDPS.	12.1, 12.2	Essential
12	Check the exit status of the PGE and examine the PGE log files for errors or anomalous messages.	22.1.1	Essential
13	Compare the output produced with test data delivered with the PGE.	15.1, 15.2, 15.3, 15.4	Essential
14	Examine the output products using data visualization techniques.	16	Optional
15	Verify that MCFs (Source MCFs) have content compatible with the granule level metadata sections of the corresponding ESDT descriptor files. If not compatible, update the descriptor files.	13.2.1, 13.1.1	Critical
16	<p>Verify that all required ESDTs have either:</p> <ul style="list-style-type: none"> (1) Been successfully registered within the ECS, or (2) Can be registered with existing descriptor files <p>If an ESDT has not been registered, then register it.</p> <p>ESDTs are needed for:</p> <ul style="list-style-type: none"> (1) All input and output data granules (2) The PGE executable tar file (SSEP) 	<p>13.1.1</p> <p>6.3</p>	Essential

Table 3.1-1. Road Map of SSI&T for a PGE (cont.)

Step	Task	Sections	Importance
17	<p>For each ESDT used by the PGE, construct an ESDT ODL file for updating the PDPS or verify that they already exist.</p> <p>ESDT ODL files are needed for:</p> <ul style="list-style-type: none"> (1) All input and output data granules (2) All MCFs (3) The PGE executable tar file (SSEP) <p>This begins PGE registration.</p>	13.1.1	Critical
18	<p>Construct a PGE ODL file for updating the PDPS database. This involves using the updated PCF to construct an initial PGE ODL file. The PGE ODL file must then be hand edited to add required metadata.</p> <p>A mapping between logical IDs in the PCF and ESDT ShortNames must be known <i>before</i> this step is done.</p>	13.1.2	Critical
19	<p>Supply operational metadata to the PDPS database.</p> <p>This completes the PGE registration.</p>	13.1.3	Critical
20	Build the PGE linking it to the DAAC version of the SDP Toolkit.	11.5	Critical
21	For each input dynamic data granule needed by the PGE, construct a Target MCF and Insert it to the Science Data Server.	13.2.1, 13.2.2	Critical
22	<p>For each input static data granule, construct a Target MCF and Insert it to the Science Data Server.</p> <p>Such a Target MCF is needed in order to Insert:</p> <ul style="list-style-type: none"> (1) All input static data granules (2) All MCFs (3) The PGE executable tar file (SSEP) 	13.2.3, 13.2.4	Critical
23	Assemble the SSEP (as a tar file) and Insert it to the Science Data Server.	17.1, 17.2	Critical
24	Initiate a Production Request (PR) that will result in a single Data Processing Request (DPR), that is, run once.	14.2	Critical

Table 3.1-1. Road Map of SSI&T for a PGE (cont.)

Step	Task	Sections	Importance
25	Register a subscription for the PGE input and output files.	13.1	Critical
26	Use the Planning Workbench to plan the PR and hence, run the PGE.	14.2	Essential
27	Monitor the PGE's progress using AutoSys.	14.3	Essential
28	Examine the Production History (PH) including the PGE log files for errors or anomalous messages.	14.4	Essential
29	Compare the output produced with test data delivered with the PGE.	15.1, 15.2, 15.3, 15.4	Essential
30	Examine the output products using data visualization techniques.	16	Optional

3.2 Set Up of SSI&T User Accounts

During EGS testing at the Goddard DAAC and early SSI&T of MODIS PGEs, much has been learned about the setting up SSI&T user accounts on the Pre-Release B Testbed. This knowledge has been used to informally develop a number of set up scripts which allow SSI&T users to have the proper environments on all machines accessed during SSI&T.

These set up scripts are under informal CM (using RCS) and are available as a compressed UNIX tar file at:

ftp://ecsinfo.hitc.com/Testbed/ssit_account_setup.tar.Z

A README file is included in the tar archive.

These files are available as informational only. There is no assumption in these Green Book procedures that they are being used.

3.3 How to Use The Procedures

The science software I&T operational procedures contained within this document are ordered. The order is intended to *loosely* suggest a logical sequence which, when used as a “road map”, represents an overall, sensible end-to-end SSI&T activity at the Release B Testbed DAACs. The ordering cannot, however, be interpreted as a detailed, step-by-step guide to SSI&T activities. In addition, since there are many factors that affect the actual SSI&T activities (*e.g.* Instrument Team deliveries, DAAC policies, agreements between the Instrument Teams and the DAACs), the ordering in this document can only be suggestive.

Each procedure document is broken up into three sections. The first section contains the Activity Checklist. This is a table containing, in broad terms, the steps necessary for carrying out the procedure. It is intended to give an overview.

The second section describes each of these steps in more detail. This section is geared to someone unfamiliar with the procedure or someone needing verbose descriptions. Conventions used in this section are as follows: Text shown in **bold** represents what is to be typed in literally via the keyboard. Text shown in ***bold italics*** represents something to be filled in by something appropriate. In all cases, the carriage return or “enter” key is represented as **Return**. For example:

cd *MyDir*, press **Return**.

means type “cd” followed by a directory name which is appropriate to the task, and followed by pressing the return key.

Text shown in **bold Helvetica** font represents names, labels, buttons, etc. on graphical user interfaces (GUIs). For example:

In the GUI labeled **Process Control File Checker**, click on the **OK** button.

The third section contains the Quick Step Procedures. This table is geared to someone already familiar with the procedure and shows a condensed view of the steps required. Experienced users can jump directly to this section to quickly be reminded of the steps.

This page intentionally left blank.

4. Pre-Release B Testbed and B.0 Architectural Overview: Impact on SSI&T Procedures

The process of SSI&T or integration of EOS Instrument Science Software into the ECS has been developed and refined over three iterations of ECS. IR1, the Pre-Release B Testbed and Release B.0. The latter will be the at-launch system supporting EOS-AM1 instruments. Although every attempt has been made to keep integration procedures for science software consistent through succeeding releases, basis architectural differences have led to significant variances. This provides a synopsis of the architecture of the Pre-Release B Testbed and of the Release B ECS in Sections 4.1 and 4.2, respectively. A list of major SSI&T procedural variations is given in Section 4.3.

4.1 Pre-Release B Testbed Architecture: Overview

The Pre-Release B Testbed was made operational at GSFC, EDC and NSDIC DAACS primarily for the purpose of conducting SSI&T for EOS instrument team Version 1 software. Several software components were included, in order to smooth the progression to Version 2 integration into ECS:

- A Planning and Data Processing System (PDPS) provided SSI&T support for registering PGEs, setting up subscriptions for Planning, entering production requests, developing and activating a production plan, scheduling and monitoring jobs with the AutoSys COTS, managing PGE execution, archiving to data storage, and QA monitoring of products including use of EOSVIEW for visualization.
- A flat file data base, a.k.a. the Integrated Metastore Factory or IMF, which were interfaced with Planning and Data Processing components. This provided a ‘data-server-like’ interface, without which, Testbed integration procedures would bear little resemblance to later B.0 procedures. The IMF supported metadata validation, supported subscription registration and notification, and allowed ESDT additions via a GUI interface.
- An “out-of-the-box” version of HP OpenView was provided for monitoring hardware failures.

From the viewpoint of SSI&T, the most critical cause of procedural differences between the Testbed and Release B.0 is the IMF. The IMF is a UNIX file system based archive simulator that provides basic science data management functionality. The system is not a server or a persistent daemon process. Rather, it was designed as a client-only solution with a standard B.0 ECS Data Server/Client interface. The system supported interfaces for inserting and retrieving science data and any associated ancillary products. Temporal based query and metadata inspect/update operations were also supported. The system also had a simplified file-based asynchronous notification mechanism for data insert events.

The IMF archive was based on a UNIX file system. Each ESDT consists of a directory and a descriptor file that describes the ESDT. Data granules were stored in each ESDT directory at

lower level in the structure. Each directory then, represented a collection. Also present in the archive was a data dictionary file that contained a superset of all valid ESDT attributes. The descriptor and the data dictionary were processed during data insert operation for metadata validation purposes. For asynchronous notification of data insert events, a data handle otherwise known as the Universal Reference (UR) file was written to a built-in repository directory.

The implementation of the IMF as a flat file system, while allowing ECS SSI&T procedures to be conducted on Science PGEs, implies major procedural differences in SSI&T for Release B.0. These differences are listed in Section 4.3.

4.2 Release B.0 Architecture: Overview

The Release B.0 architecture can be grouped into the following four categories:

- Data storage and management is provided by the Data Server Subsystem (DSS), with the functions needed to archive science data, search for and retrieve archived data, manage the archives, and stage data resources needed as input to science software or resulting as output from their execution. The Data Server Subsystem provides access to earth science data in an integrated fashion through an Application Programming Interface (API) that is common to all layers.
- Information search and data retrieval is provided by the science user interface functions in the Client Subsystem (CLS), by information search support functions in the Data Management Subsystem (DMS), and by capabilities in the Interoperability Subsystem (IOS) which assist users in locating services and data. These subsystems comprise what is referred to as the “Pull” side of ECS.
- Data processing is provided by the Data Processing Subsystem (DPS) for the science software, and by capabilities for long and short term planning of science data processing, as well as by management of the production environment provided by the Planning Subsystem (PLS). Routine data processing and re-processing will occur in accordance with the established production plans. In addition ECS will provide "on-demand processing", where higher level products are produced only when there is explicit demand for their creation.
- Data ingest is provided by the Ingest Subsystem (INS), which interfaces with external applications and provides data staging capabilities and storage for an approximately 1-year buffer of Level 0 data (so that reprocessing can be serviced from local storage). The INS is designed to accommodate a potentially large number of external interfaces. It is also designed to provide very diverse functions, such as high-volume ingest of level 0 data and low-volume ingest of data from field campaigns. Data Ingest and data processing comprise what is known as the “Push” side of ECS.

Figure 4.1 provides a context diagram for ECS B.0 subsystems.

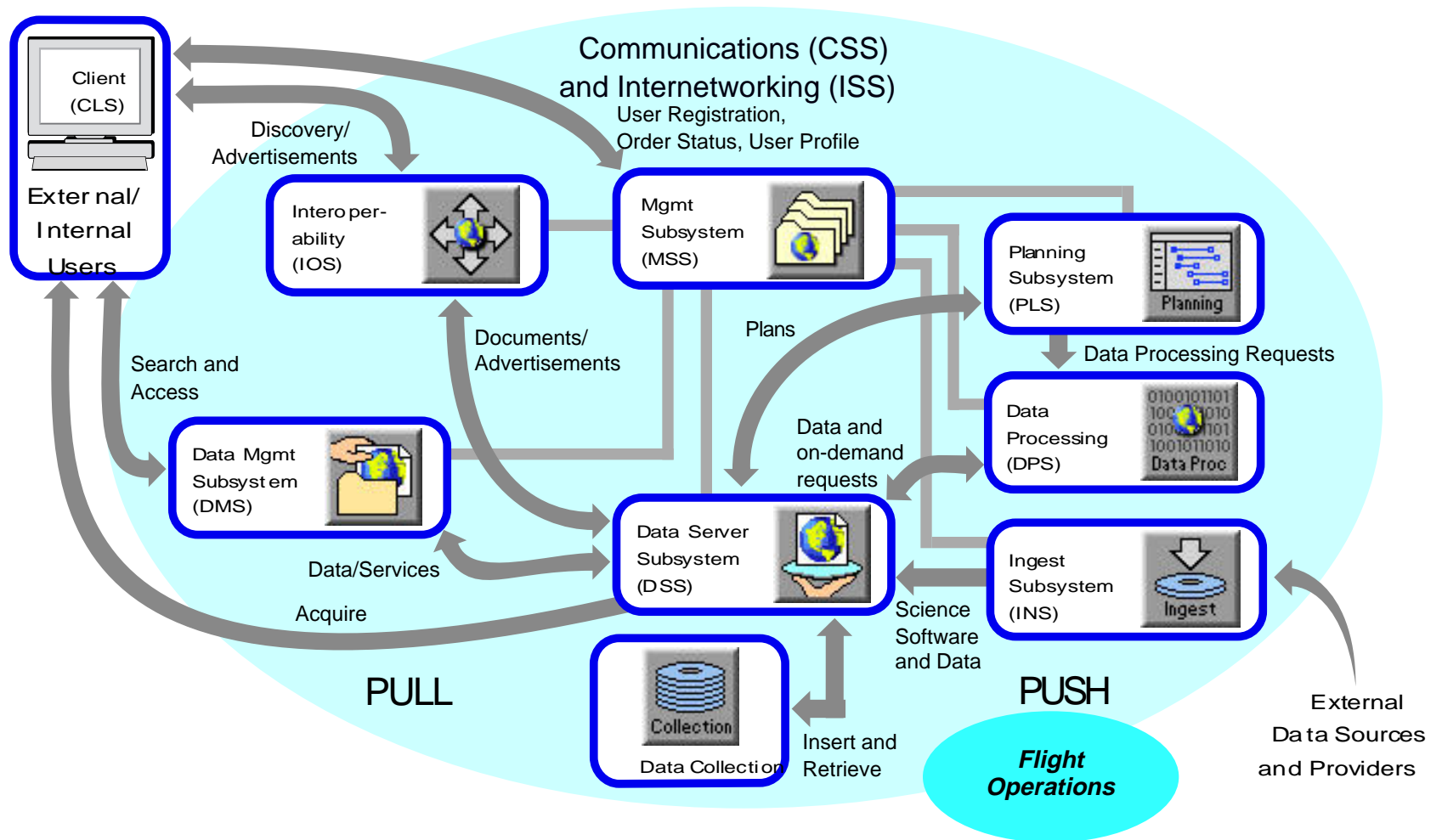


Figure 4.1. B.0 Context Diagram

The following sub-sections provide brief overviews for each of these subsystems. More detailed discussions of their design breakdown can be found the *Release-B SDKs/CSMS Design Specification Overview*, 305-CD-020-002.

4.2.1 Client Subsystem (CLS)

The Client provides users with an interface via which they can access ECS services and data. It also gives science software access to the ECS services, as well as direct access to ECS data. Access is provided through graphic user interface (GUI) application tools for displaying the various kinds of ECS data (e.g., images, documents, tables), and libraries representing the client APIs to ECS services. The client subsystem follows an object oriented design. The design is built around a core set of 'root' objects from which all other software will inherit its behavior.

4.2.2 Interoperability Subsystem (IOS)

The Interoperability subsystem provides an advertising service. It maintains a database of information about the services and data offered by ECS, and provides interfaces for searching this database and for browsing through related information items. For example, ESDTs, are made visible through the advertising service. ESDTs are objects which contain metadata describing data granules and the services which the system can apply to those granules. The Client Subsystem provides the user interface which enables access to the IOS.

4.2.3 Data Management Subsystem (DMS)

The Data Management subsystem provides three main functions:

- Provide end-users with a consolidated logical view of a distributed set of data repositories.
- Allow end-users to obtain descriptions for the data offered by these repositories. This also includes descriptions of attributes about the data and the valid values for those attributes.
- Provide data search and access gateways between ECS and external information systems.

4.2.4 Data Server Subsystem (DSS)

The Data Server subsystem provides the management, cataloging, access, physical storage, distribution functions for the ECS earth science data repositories, consisting of science data and their documentation. The Data Server provides interfaces for other ECS subsystems which require access to data server services. The Data Server Subsystem consists of the following principal design components:

- Database Management System - The Data Server subsystem will use database technology to manage its catalog of earth science data, and for the persistence of its system administrative and operational data.
- Document Management System - Web server and database technology are used to implement a document management system to provide storage and information retrieval for guide documents, science software documentation, and ECS earth science related documents.
- Data Type Libraries - The Data Server will use custom dynamic linked libraries (DLLs) to provide an means of implementing the variety of ECS earth science data types and services, and will provide a interface for use by other ECS subsystems requiring access to those services and data.
- File Storage Management System - This component provides archival and staging storage for data.
- Distribution System - The Data Server provides the capabilities needed to distribute bulk data via electronic file transfer or physical media.

4.2.5 Ingest Subsystem (INS)

This subsystem handles the initial reception of all data received at an ECS facility and triggers subsequent archiving and processing of the data. The ingest subsystem is organized into a collection of software components (e.g., ingest management software, translation tools, media handling software) from which those required in a specific situation can be readily configured. The resultant configuration is called an ingest client. Ingest clients can operate on a continuous basis to serve a routine external interface; or they may exist only for the duration of a specific ad-hoc ingest task. The ingest subsystem also standardizes on a number of possible application protocols for negotiating an ingest operation, either in response to an external notification, or by polling known data locations for requests and data.

4.2.6 Data Processing Subsystem (DPS)

The main components of the data processing subsystem - the science algorithms or Product Generation Executives (PGEs) - will be provided by the science teams. The data processing subsystem provides the necessary hardware resources, as well as a software environment for queuing, dispatching and managing the execution of these algorithms. The processing environment will be highly distributed and will consist of heterogeneous computing platforms. The AutoSys COTS tool is used as the scheduling engine. The tool is designed to manage production in a distributed UNIX environment. The DPS also interacts with the DSS to cause the staging and de-staging of data resources in synchronization with processing requirements.

4.2.7 Planning Subsystem (PLS)

The Planning Subsystem provides the functions needed to plan routine data processing, schedule on-demand processing, and dispatch and manage processing requests. The subsystem provides

access to the data production schedules at each site, and provides management functions for handling deviations from the schedule to operations and science users. The Planning subsystem provides several functions to account for:

- A processing environment which will be highly distributed and consist of heterogeneous computing platforms.
- Existence of inter-site and external data dependencies.
- Dynamic nature of the data and processing requirements of science algorithms.
- High system availability.
- Provision of a resource scheduling function which can accommodate hardware technology upgrades.
- Support for on-demand processing (as an alternative to predominantly routine processing).
- Ability to provide longer-term (e.g., monthly) processing predictions as well as short term (e.g., daily) planning and scheduling

4.2.8 Communications Subsystem (CSS)

The CSS helps manage the operation of distributed objects in ECS, by providing a communications environment. The environment allows software objects to communicate with each other reliably, synchronously as well as asynchronously, via interfaces that make the location of a software object and the specifics of the communications mechanisms transparent to the application.

In addition, CSS provides the infrastructural services for the distributed object environment. They are based on the Distributed Computing Environment (DCE) from the Open Software Foundation (OSF). DCE includes a number of basic services needed to develop distributed applications, such as remote procedure calls (rpc), distributed file services (DFS), directory and naming services, security services, and time services.

Finally, CSS provides a set of common facilities, which include legacy communications services required within the ECS infrastructure and at the external interfaces for file transfer, electronic mail, bulletin board and remote terminal support. The Object Services support all ECS applications with interprocess communication and specialized infrastructural services such as security, directory, time, asynchronous message passing, event logging, lifecycle service, transaction processing and World Wide Web (WWW) service.

4.2.9 Management Subsystem (MSS)

The Management Subsystem (MSS) provides enterprise management (network and system management) for all ECS resources: commercial hardware (including computers, peripherals, and

network routing devices), commercial software, and custom applications. With few exceptions, the management services are decentralized, such that no single point of failure exists.

MSS provides two levels of an ECS management view: the local (site/DAAC specific) view, provided by Local System Management (LSM), and the enterprise view, provided by the Enterprise Monitoring and Coordination (EMC) at the SMC. Enterprise management relies on the collection of information about the managed resources, and the ability to send notifications to those resources. For network devices, computing platforms, and some commercial off-the-shelf software, MSS relies on software called "agents" which are usually located on the same device/platform and interact with the device's or platform's control and application software, or the commercial software product. However, a large portion of the ECS applications software is custom developed, and some of this software - the science software - is externally supplied. For these components, MSS provides a set of interfaces via which these components can provide information to MSS (e.g., about events which are of interest to system management such as the receipt of a user request or the detection of a software failure). These interfaces also allow applications to accept commands from MSS, provided to MSS from M&O consoles (e.g., an instruction to shut down a particular component). Applications which do not interact with MSS directly will be monitored by software which acts as their "proxies". For example, the Data Processing Subsystem (DPS) acts as the proxy for the science software it executes. DPS notifies MSS of events such as the dispatching or completion of a PGE, or its abnormal termination.

MSS uses HP OpenView as its system management tool. The information collected via the MSS interfaces from the various ECS resources is consolidated into an event history database, some on a near real-time basis, some on a regular polling basis (every 15-to 30 minutes), as well as on demand, when necessitated by an operator inquiry. The database is managed by Sybase, and Sybase query and report writing capabilities are used to extract regular and ad-hoc reports from it. Extracts and summaries of this information will be further consolidated on a system wide basis by forwarding it to the SMC (also on a regular basis).

MSS provides fault and performance management and other general system management functions such as security management (providing administration of identifications, passwords, and profiles); configuration management for ECS software, hardware, and documents; Billing and Accounting; report generation; trending; request tracking; and mode management (operational, test, simulation, etc.).

4.2.10 Internetworking Subsystem (ISS)

The ISS provides local area networking (LAN) services at ECS installations to interconnect and transport data among ECS resources. The ISS includes all components associated with LAN services including routing, switching, and cabling as well as network interface units and communications protocols within ECS resources.

The ISS also provides access services to link the ECS LAN services to Government-furnished wide-area networks (WANs), point-to-point links and institutional network services. Examples include the NASA Science Internet (NSI), Program Support Communications Network (PSCN), and various campus networks "adjoining" ECS installations.

4.3 Implications for SSI&T Procedures.

The Pre-Release B Testbed was used as an integration environment, for EOS instrument team Version 1 software, from May, 1997 through November, 1997. The Release B.0 system will be used after January, 1998 to integrate Version 2, or launch-ready instrument processing software for EOS-AM1. Architectural differences between the two ECS releases will cause certain SSI&T procedural differences. The major differences are due to the presence of Ingest and Data Server Subsystems in B.0. Table 4.1 list the major differences in procedures, which will be encountered by SSI&T staff, in integrating Science Version 1 and Version 2 software.

Table 4.1. Major SSI&T Procedural Differences: Testbed to B.0 Releases

Function	Pre-Release B Testbed	Release B.0
System Operation	Run custom binaries, COTS (AutoSys); no servers. non-DCE	All servers must run and communicate with each other; bring up manually, or use ECSAssistant tool.
Ingest Ancillary Data Granules	SSI&T manager tool, EDSTs visible in IMF	Ingest GUI, ESDTs must be visible to IOS server.
ESDT Insert	SSI&T manager tool	Use Ingest tool
ESDT Verification	View of flat file database	Verify through IOS
DAP, SSAP Insert	Direct copy	Use Ingest
PDPS Database Population	No production rules	More attributes, production rules
PGE Operation	Activate from inside AutoSys, operator choice automatic reprocessing facility for failed PGE. simple chaining (e.g. time ordered)	When all data is available; DPR activated. No automatic reprocessing Complex chaining through production rules.
File Access	Copy from IMF, single site	Verify presence through IOS; ftp from SDSRV; access to multiple sites
Multi-file Granule Support	Not supported directly, files inserted separately, accessed by work-around	Files inserted together, accessed as a single granule
Subscription Management	Reaper - non-DCE version	Subscription Manager

5. Configuration Management

This section describes procedures for handling the configuration management of science software delivered to the DAACs for SSI&T. The COTS tool used for this purpose is ClearCase® by Atria Software, Inc. ClearCase can be run from the command line and via a graphical user interface (GUI).

All data managed under ClearCase are stored in VOBs (Versioned Object Bases), which are the “public” storage areas and views, which are the “private” storage areas. VOBs are data structures that can only be created by a VOB administrator using the mkvob (“make vob”) command. They are mounted as a file system and when viewed through a view, VOBs appear as standard UNIX directory tree structures. This file system, accessed through its mount point, has a version-control dimension which contains file elements and versions of file elements. ClearCase maintains multiple versions of a file organized into a hierarchical version tree which may include many branches. For the purposes of SSIT, it is suggested that branches not be created in managing the files of the science software. All versions of the software files should remain on their main branch of the tree and therefore easily accessed by the default configuration of any view that may be set by the user.

Data that are under configuration management in ClearCase are said to be *checked in*. In order to alter a checked in data element (*e.g.* a file) to make a newer version of it, the data element must first be *checked out*. Once the change has been made to the checked out version, it is checked in again. The VOB will then contain both versions of the data element (in this case, a file) and either can be retrieved at a later time.

In general, executable binary files, object files, and data files should not be checked into ClearCase. Binary and object files are not stored efficiently in ClearCase; data files for science software may be extremely large (multiple gigabytes) and a VOB is typically not sized for this. Files that should be checked into ClearCase include source code, scripts, makefiles, assorted build and run scripts, documentation, and other ASCII files.

The administrator in charge of the VOB is referred to as the VOB administrator (VA).

All ClearCase procedures assume that the user’s umask is set to 002.

5.1 Creating a View in ClearCase

In order to make files and directories that are in a ClearCase VOB (Versioned Object Base) visible and accessible, a ClearCase view must be set. This procedure describes how to create a ClearCase view to be used later (in Section 6.2).

A ClearCase view need only be created once. Once created, the view can be set at the beginning of each user session. Multiple views for a single user may be created.

In order for the SSI&T tools under the SSIT Manager to have access to the ClearCase VOB, the ClearCase view must be set *before* the SSIT Manager is run.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ClearCase VOB has been properly created and populated.
2. The VA has granted ClearCase privileges for the user.

To create a ClearCase view, execute the procedure steps that follow:

- 1 At a UNIX prompt, type **cleartool mkview -tag *ViewName* *ViewPath/ViewName.vws***, press **Return**.
 - The ***ViewPath*** is the full path name to the directory where ClearCase views are stored. This information should be supplied by the SA. A typical example is `/net/mssg1sungsfsc/viewstore/`.
 - The ***ViewName*** is the name of the view being created (typically set to the user id, for example, `jdoe`). View names should have `.vws` as the file name extension.
 - For example, to create a view for user `jdoe`, type **cleartool mkview -tag *jdoe* /net/mssg1sungsfsc/data/viewstore/*jdoe.vws***, press **Return**.
 - Multiple views can be created in the same manner for the same user. Each view, however, must have a unique name. For example, `jdoe_test` and `jdoe_devel`.

Table 5.1-1. Creating a View in ClearCase - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	cleartool mkview -tag <i>ViewName</i> <i>ViewPath/ViewName.vws</i>	press Return

5.2 Setting a View in ClearCase

In order to make files and directories that are in a ClearCase VOB (Versioned Object Base) visible and accessible, a ClearCase view must be set. Only one view can be set (active) at a time.

Section 6.1 describes how to create a ClearCase view.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ClearCase VOB has been properly created and populated.
2. A ClearCase view has been created for the user (see Section 6.1).

To set a ClearCase view, execute the procedure steps that follow:

- 1 At a UNIX prompt, type **cleartool setview *ViewName***, press **Return**.
 - The ***ViewName*** is the name of the view to be set. For example, type **cleartool setview jdoe**, press **Return**.
 - The ***ViewName*** may be a view that another user has created and owns. The restriction, in this case, is that files cannot be checked in.

Table 5.2-1. Setting a View in ClearCase - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	cleartool setview <i>ViewName</i>	press Return

5.3 Importing Multiple Files into ClearCase from a Directory Structure

This procedure explains how to place the entire contents of a UNIX directory structure under ClearCase. A UNIX directory structure refers to all the files and subdirectories under some top-level directory.

This procedure is geared toward science software deliveries. In such cases, science software is delivered in the form of a UNIX *tar* files. A *tar* file has been unpacked (untar-red) and the contents are to be placed under ClearCase configuration management.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ClearCase VOB has been properly created.
2. A ClearCase view is **not** required to perform this procedure.

To import multiple files and/or directories into ClearCase from a UNIX directory structure, execute the procedure steps that follow:

- 1 At a UNIX prompt, type **cd *ParentPathname***, press **Return**.
 - The ***ParentPathname*** is the path name of the directory that *contains* the directory structure to be brought into ClearCase. This is *not* the VOB. For example, to bring the pge2 directory which sits under /MOPITT (i.e. its path name is /MOPITT/pge2) into ClearCase along with all files and subdirectories below it, **cd /MOPITT**, press **Return**.
- 2 At the UNIX prompt, type **clearcvt_unix -r *DirName***, press **Return**.
 - The ***DirName*** is the name of the directory in which it and everything below it is to be brought into ClearCase. For example, based on the example in step 1, **clearcvt_unix -r pge2**, press **Return**. A conversion script will be

then be created. The **-r** causes all subdirectories to be recursively included in the script created.

- 3 Contact the VA and request that the utility script **cvt_script** be run on the script created in step 2.
 - The VOB administrator (VA) is the only one who can run the **cvt_script** because it modifies the VOB.

Table 5.3-1. Importing Multiple Files into ClearCase from a Directory Structure - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<code>cd ParentPathname</code>	press Return
2	<code>clearcvt_unix -r DirName</code>	press Return
3	(No entry)	request VA run <code>cvt_script</code>

5.4 Entering a Single File into ClearCase

This procedure explains how to put a single file under configuration management using ClearCase.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ClearCase VOB has been properly created.
2. A ClearCase view exists for the user through which the desired VOB directory can be seen.

To enter a single file into ClearCase, execute the procedure steps that follow:

- 1 At a UNIX prompt, type **cleartool setview ViewName**, press **Return**
 - The **ViewName** is the name of the ClearCase view (see Section 6.1 to create a view). For example, **cleartool setview jdoe**, press **Return**.
- 2 At the UNIX prompt, type **cd pathname**, press **Return**.
 - The **pathname** is the full path name of the subdirectory in the VOB into which the file is to be checked in. For example, to check a file into the VOB directory `/VOB1/pge2/scripts/`, type **cd /VOB1/pge2/scripts/**, press **Return**. If the desired directory cannot be seen, it could mean that the view has not been set or the properties of the view do not allow the directory to be seen; check with the VA.

- 3 At a UNIX prompt, type **cp *pathname/filename* .**, press **Return** (note the space and then “dot” at the end of the command).
 - The ***pathname*** is the full path name to the directory where the file to be checked in exists and ***filename*** is the file name of the file to be checked in. This command copies a file over into the VOB area in preparation for checking it in. For example, to copy over a file named MISR_calib.c in directory /pge/pge34/ to be checked in, type **cp pge/pge34/MISR_calib.c .**, press **Return** (again, note the space and then “dot” at the end of the command).
- 4 At the UNIX prompt, type **cleartool checkout -nc .**, press **Return** (note the space and then “dot” at the end of the command).
 - This command checks out the current directory (represented by the “dot”) from ClearCase. Adding a new file (or element) to a directory represents a modification of the directory. Hence, the directory must be checked out before a file can be checked in.
- 5 At a UNIX prompt, type **cleartool mkelem -nc *filename***, press **Return**.
 - The ***filename*** is the name of the file that was copied over in step 4 and is the file that will be checked into ClearCase. This command creates a ClearCase element from the file in preparation for checking it in. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the make element step.
- 6 At the UNIX prompt, type **cleartool checkin -nc *filename***, press **Return**.
 - The ***filename*** is the name of the file to be checked into ClearCase. This command performs the check in of the file. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check in step.
- 7 At the UNIX prompt, type **cleartool checkin -nc .**, press **Return** (note the space and then “dot” at the end of the command).
 - This command checks in the current directory (represented by the “dot”) into ClearCase. The adding of an element (here, a file) represents a modification to the directory and hence, the new version of the directory must be checked back in. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check in step.

Table 5.4-1. Entering a Single File into ClearCase - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<code>cleartool setview <i>ViewName</i></code>	press Return
2	<code>cd <i>pathname</i></code>	press Return
3	<code>cp <i>pathname/filename</i> .</code>	press Return
4	<code>cleartool checkout -nc .</code>	press Return
5	<code>cleartool mkelem -nc <i>filename</i></code>	press Return
6	<code>cleartool checkin -nc <i>filename</i></code>	press Return
7	<code>cleartool checkin -nc .</code>	press Return

5.5 Entering a New Directory into ClearCase

This procedure explains how to put a new directory (empty) into ClearCase.

The assumptions are that a VOB exists and is mounted at a known UNIX directory. A ClearCase view exists for the SSI&T operator.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ClearCase VOB has been properly created.
2. A ClearCase view exists for the user through which the desired VOB parent directory can be seen.

To enter a single file into ClearCase, execute the procedure steps that follow:

- 1 At a UNIX prompt, type **cleartool setview *ViewName***, press **Return**
 - The ***ViewName*** is the name of the ClearCase view (see Section 6.1 to create a view). For example, **cleartool setview *jdoe***, press **Return**.
- 2 At the UNIX prompt, type **cd *pathname***, press **Return**.
 - The ***pathname*** is the full path name of the parent directory in the VOB in which the new directory is to be added. For example, if a new directory is to be added under **/VOB1/pge4**, type **cd /VOB1/pge4**, press **Return**.
- 3 At the UNIX prompt, type **cleartool checkout -nc .**, press **Return** (note the space and then “dot” at the end of the command).
 - This command checks out the current directory (represented by the “dot”) from ClearCase. This directory will be the parent directory of the new directory. Adding a new directory (or element) to another directory represents a modification of the directory. Hence, the directory must be checked out before a new directory can be added and checked in.
- 4 At a UNIX prompt, type **cleartool mkdir -nc *DirectoryName***, press **Return**.

- The ***DirectoryName*** is the name of the new directory being created. This command creates the new directory.
 - The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the directory creation step.
- 5 At the UNIX prompt, type **cleartool checkin -nc *DirectoryName***, press **Return**.
- The ***DirectoryName*** is the name of the new directory created in step 4. This command performs the check in of the new directory. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check in step.
- 6 At the UNIX prompt, type **cleartool checkin -nc .**, press **Return** (note the space and then “dot” at the end of the command).
- This command checks in the current directory (represented by the “dot”) into ClearCase. The adding of an element (here, a directory) represents a modification to the current directory and hence, the new version of the directory must be checked back in.
 - The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check in step.

Table 5.5-1. Entering a New Directory into ClearCase - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	cleartool setview <i>ViewName</i>	press Return
2	cd <i>pathname</i>	press Return
3	cleartool checkout -nc .	press Return
4	cleartool mkdir -nc <i>DirectoryName</i>	press Return
5	cleartool checkin -nc <i>DirectoryName</i>	press Return
6	cleartool checkin -nc .	press Return

5.6 Checking Out an Element From ClearCase

This procedure explains how to check out an element from ClearCase so that it can be modified. An element refers to a directory or file in ClearCase, that is, under configuration management. Modifications made to a file or directory cannot be saved in ClearCase unless the file or directory had been checked out first.

The assumptions are that a VOB exists and is mounted at a known UNIX directory. A ClearCase view exists for the SSI&T operator.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ClearCase VOB has been properly created.

2. A ClearCase view exists for the user through which the desired VOB element can be seen. See Section 6.7 for how to check in a modified element.

To enter a single file into ClearCase, execute the procedure steps that follow:

- 1 At a UNIX prompt, type **cleartool setview *ViewName***, press **Return**
 - The ***ViewName*** is the name of the ClearCase view (see Section 6.1 to create a view). For example, **cleartool setview jdoe**, press **Return**.
- 2 At the UNIX prompt, type **cleartool checkout -nc *element***, press **Return**.
 - The ***element*** is the name of the file or directory (full path name allowed) that is to be checked out (and later modified). The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check out step.
- 3 This step is optional; it is performed to cancel a check out. At a UNIX prompt, type **cleartool uncheckout -nc *element***, press **Return**.
 - The ***element*** is the name of the file or directory (full path name allowed) checked out (as in step 2, above). This command cancels the check out of an element. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the uncheck out step. Note that ClearCase will not allow an unmodified element to be checked in. Instead, the uncheck out command must be used.

Table 5.6-1. Checking Out an Element From ClearCase - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	cleartool setview <i>ViewName</i>	press Return
2	cleartool checkout -nc <i>element</i>	press Return
3	(Optional) cleartool uncheckout -nc <i>element</i>	press Return

5.7 Checking a Modified Element into ClearCase

This procedure explains how to check in a modified element to ClearCase. An element refers to a directory or file in ClearCase, that is, under configuration management. Modifications made to a file or directory cannot be saved in ClearCase unless the file or directory had been checked out first. See Section 6.6 on how to check out an element.

The assumptions are that a VOB exists and is mounted at a known UNIX directory. A ClearCase view exists for the SSI&T operator.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ClearCase VOB has been properly created.
2. A ClearCase view exists for the user through which the desired VOB element can be seen. See Section 6.6 for how to check out an element before modifying.

To enter a single file into ClearCase, execute the procedure steps that follow:

- 1 At a UNIX prompt, type **cleartool setview *ViewName***, press **Return**.
 - The ***ViewName*** is the name of the ClearCase view (see Section 6.1 to create a view). For example, **cleartool setview *jdoe***, press Return.
- 2 At the UNIX prompt, type **cleartool checkin -nc *element***, press **Return**.
 - The ***element*** is the name of the file or directory (full path name allowed) that is to be checked out (and later modified). The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check out step.
- 3 This step is optional; it is performed when ClearCase does not accept a check in because the element was not modified. In this case, the check out must be canceled. At a UNIX prompt, type **cleartool uncheckout -nc *element***, press **Return**.
 - The ***element*** is the name of the file or directory (full path name allowed) checked out. This command cancels the check out of an element. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the uncheck out step.

Table 5.7-1. Checking a Modified Element into ClearCase - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	cleartool setview <i>ViewName</i>	press Return
2	cleartool checkin -nc <i>element</i>	press Return
3	(Optional) cleartool uncheckout -nc <i>element</i>	press Return

This page intentionally left blank.

6. The ECS Assistant

ECS (EOSDIS Core System) is a complex system comprising many subsystems and components running on multiple heterogeneous host machines. The coordination of all the subsystems for SSI&T is an arduous, time consuming, error prone task. In order to improve our effectiveness and efficiency, an easy-to-use GUI tool, “ECS Assistant,” has been developed to facilitate ECS SSI&T activities.

Currently, the ECS Assistant is comprised of three major scripts: EcCoAssist, EcCoModemgr, and EcCoEsdtmgr. These scripts provide users with a Graphical User Interface to perform functions such as subsystem server startup and shutdown, ESDT management, and database review when using the ECS system. During the course of performing their tasks, SSI&T operators can use ECS Assistant to perform the following functions through its GUI:

- To start up and shut down servers for each subsystem
- To graphically monitor the server up/down status
- To open and view the detailed log files for each server used
- To add, remove, and verify ESDTs for SSI&T
- To review various databases used in the ECS system

In the following sections, we will address aspects of how to use the ECS Assistant in our SSI&T activities. Section one explains how to use ECS Assistant to facilitate and manage the subsystems and their servers, including server start up and shut down. Section two describes how to monitor servers for each subsystem, including using the ECS Monitor and ECS logfile viewer. Section three contains ESDT management, which includes installing ESDTs to a mode, adding and removing an ESDT via the Science Data Server, and reviewing the Science Data Server database through the DB Viewer GUI provided by ECS Assistant.

6.1 Using ECS Assistant to Start Up / Shut Down Servers

This procedure describes routings for using the ECS Assistant GUI to start up and shut down subsystem servers. The procedure described here will apply to all the servers from different subsystems. The next procedure, in Section 6.2, will describe how to monitor the servers’ status with the ECS Assistant.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ECS Assistant has been properly installed.
2. The required environment variables have been set properly.

To run the ECS Assistant, execute the procedure steps that follow:

- 1 Log into one of the host machines used for SSIT.
- 2 At the UNIX prompt on the host from which the ECS Assistant is to be run, type **setenv DISPLAY *hostname*:0.0**, press **Return**.
 - The *hostname* is the name of the machine on which the ECS Assistant is to be displayed, *i.e.*, the machine that you are using.
 - To verify the setting, type **echo \$DISPLAY**, press **Return**.
- 3 At the UNIX prompt on the host from which the ECS Assistant is to be run, type **setenv ECS_HOME /usr/ecs**, press **Return**.
 - To verify the setting, type **echo \$ECS_HOME**, press **Return**.
- 4 If necessary, at the UNIX prompt on the host from which the ECS Assistant is to be run, type **cleartool setview *ViewName***, press **Return**.
 - The *ViewName* is the ClearCase view to be used while the ECS Assistant is running in this session. For example, type **cleartool jdoe**, press **Return**.
 - A ClearCase view is required only if the ECS Assistant needs to be able to “see” into a ClearCase VOB; a view is not necessary otherwise.
- 5 At the UNIX prompt, type **cd /tools/common/ea**, press **Return**. Then type **EcCoAssist &**, press **Return**.
 - **/tools/common/ea** is the path where ECS Assistant is installed.
 - This will invoke the ECS Assistant GUI with three push buttons for selecting the proper activities, as indicated in Fig. 6.1-1.

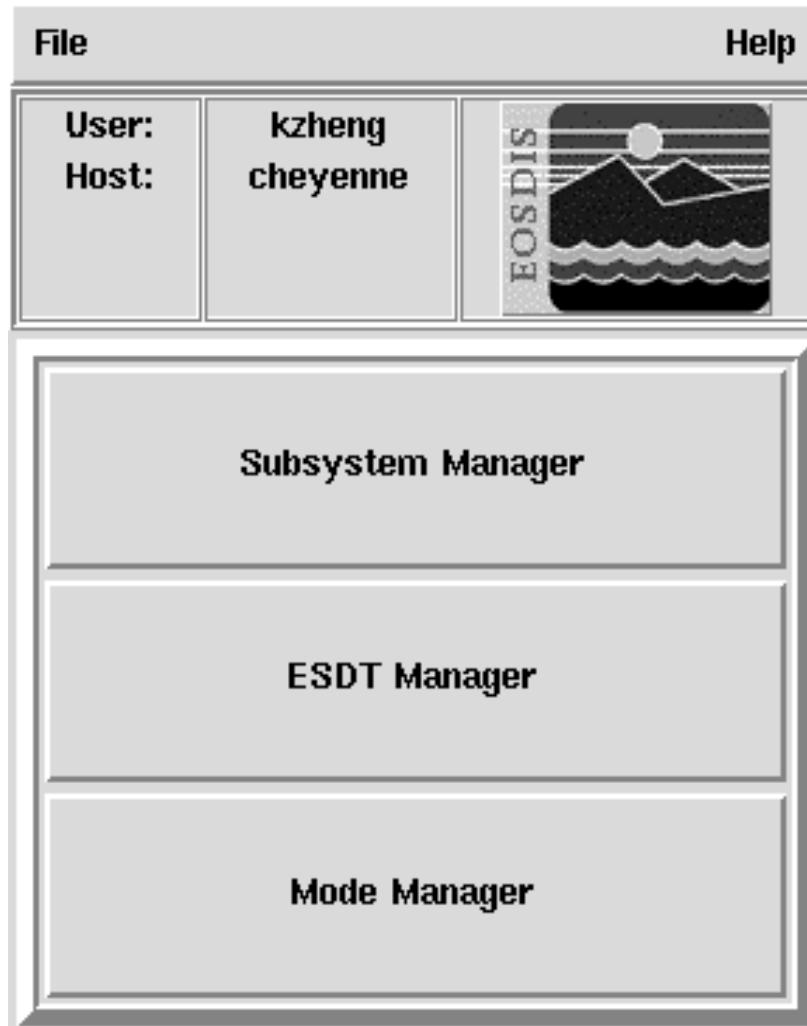


Figure 6.1-1. ECS Assistant GUI

- 6 At the ECS Assistant GUI, click the **Subsystem Manager** pushbutton.
 - This will invoke the Subsystem Manager GUI, as indicated in Fig. 6.1-2.

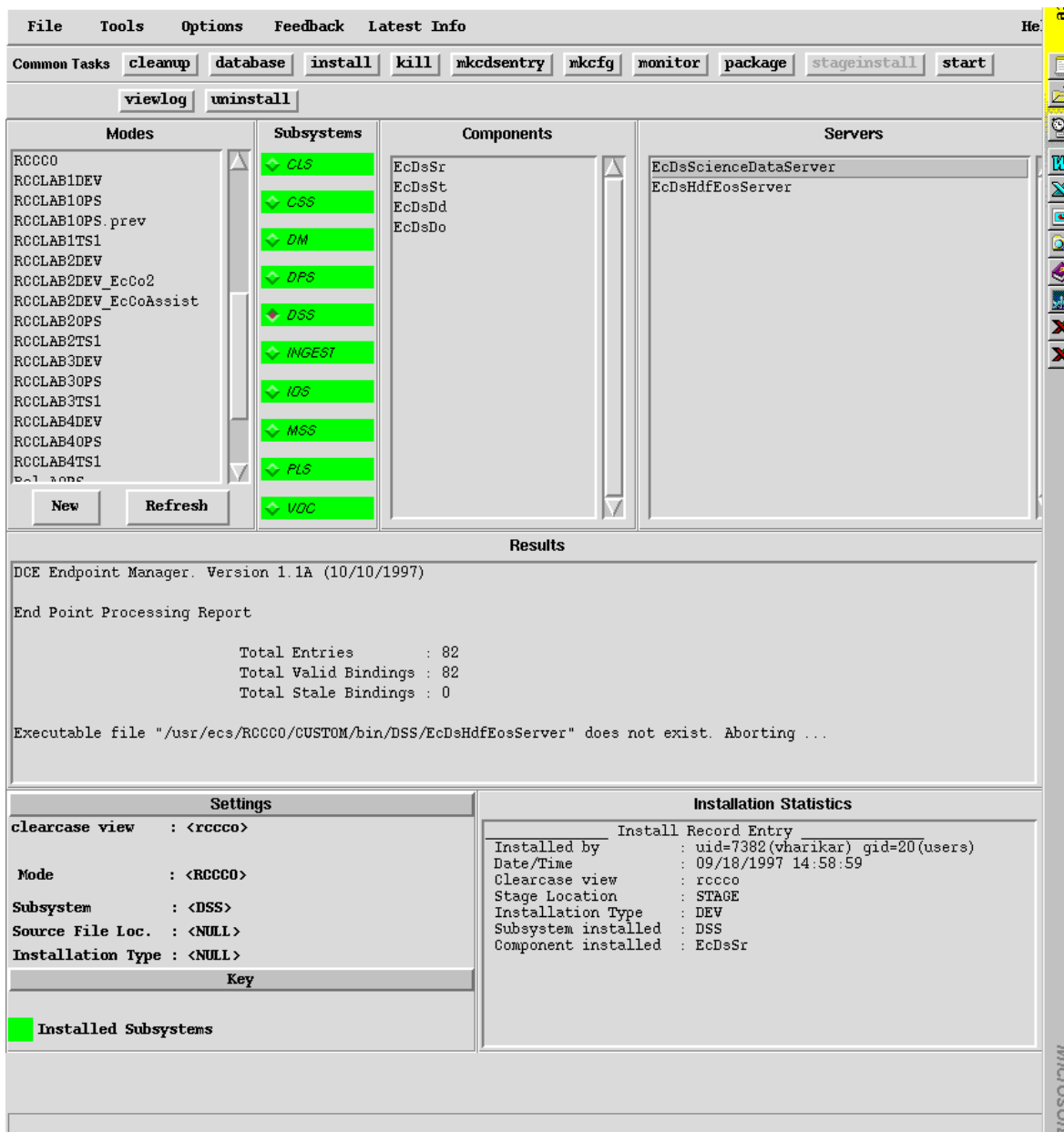


Figure 6.1-2. Subsystem Manager GUI

- 7 Select a mode by clicking a mode in the mode listing. The mode should be the one to be used for SSI&T.
 - Once the mode is selected, the color of the subsystem name list is changed.
- 8 Select a subsystem with the **Subsystem** radio button.
 - The component list for the selected subsystem will appear in the component window.

- 9 Select a component by clicking the component name under the component window.
 - The selected component will be highlighted.
 - The server list corresponding to that component will appear in the server window.
- 10 Select a server by clicking the server name from the server list under the servers window.
 - The server selected is highlighted.
- 11 To start a server up or shut it down:
 - Click the **start** button in the common tasks bar. This will start up the selected server.
 - Click the **kill** button in the common tasks bar. This will shut down the selected server.
- 12 Repeat steps 7-11 to start up or shut down other servers.
- 13 To exit the Subsystem Manager GUI, select **File..Exit** in the menu bar of the Subsystem Manager GUI.
 - This will terminate the Subsystem Manager GUI.

Table 6.1-1. Server Start Up/Shut Down Using ECS Assistant - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	(No entry)	Log onto a host
2	setenv DISPLAY <i>hostname:0.0</i>	Press Return
3	setenv ECS_HOME /usr/ecs/	Press Return
4	(If necessary) cleartool setview <i>ViewName</i>	Press Return
5	cd /tools/common/ea	Press Return
6	Type EcCoAssist	Press Return
7	invoke the Subsystem Manager GUI	Click the Subsystem Manager Push button
8	Select a mode	Click a mode name in the mode list
9	Select a component	Click a component name under the component window
10	Select a server	Click a server name under the server window
11	To start up a server	Click the start button in the common tasks bar
12	To shut down a server	Click the kill button in the tasks bar
13	Repeat steps 8 - 12 for other servers	
14	To exit	Select File..Exit

6.2 Using ECS to Perform System Monitoring

ECS Assistant provides two ways to monitor server status. The first one is by performing “tail -f” to log files which record the important activity history performed on the servers. The other way is by using a database table to display server up/down status’ dynamically. These monitoring methods are described in the following sections.

6.2.1 Using ECS Assistant to Open / View Log Files for a Selected Server

Log files are used extensively in the ECS system to record a history of activity performed on the system. They provide useful information about server activities. ECS Assistant provides an easy way to access and view these log files. In the Subsystem Manager GUI, there is one button called **viewlog** in the Common Tasks bar. Click this button to invoke a log file GUI, as shown in Fig. 6.2.1-1. You can review the log files for a particular server by choosing the server name from the Menu for the Subsystem to which it belongs. You can also view all of the log files for a component by choosing it in the Components menu. Menu entries are dimmed if no log files are present. The following example shows how to use this GUI to open log files for a particular server.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ECS Assistant has been properly installed.
2. The ECS Subsystem manager has been invoked.

To run the Log Viewer, execute the procedure steps that follow:

- 1 Click the **viewlog** button in the Subsystem Manager GUI.
 - This will invoke the log viewer GUI.
- 2 To open and view log files for a particular server, select a server from the Subsystem pull down menu, then click the server name.
 - This will open all the log files corresponding to that server.
 - The log file name is indicated in the title bar for each log file GUI.
- 3 The log file GUI provides the following options for users to view log file contents. Follow the guidance in the GUI to select the proper options:
 - **Foreground color** for changing the foreground color.
 - **Background color** for changing the background color.
 - **Font size** for changing font sizes.
 - **View entire file** for displaying the entire file.

- **Continuous update (tail -f)** for displaying the updated log file continuously.
 - **Search for** for performing word searches in the log file.
 - **Print** for printing the log file.
- 4 To view log files for other servers, repeat steps 1-3.
 - 5 Exit the log file by pressing **EXIT**.

Table 6.2.1-1. Using ECS to View the Logfile - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	To invoke the viewlog GUI	Click the viewlog button in the Subsystem Manager GUI
2	To open and view log files for a particular server	Select a server from the Subsystem pull down menu, then click the server name
3	Select option to view the log file content	Follow the GUI directions
4	To view log files for other servers	Repeat steps 1-3
5	Exit the log file GUI	Click the EXIT button

6.2.2 Using ECS Assistant to Monitor Server Status

ECS Assistant provides another convenient way to monitor the status of the servers by listing their up/down condition. The status flag for a server is up or down indicating whether or not that server is running.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ECS Assistant has been properly installed.
2. The ECS Subsystem Manager is running.

To start up the ECS monitor GUI, execute the procedure steps that follow:

- 1 At the ECS **Subsystem Manager** GUI, select a mode by clicking a mode in the mode list.
 - The mode should be the one to be used for SSI&T.
 - Once the mode is selected, the color of the subsystem name list is changed.

- 2 Select a subsystem by clicking the radio button next to the subsystem name under the subsystem component window.
 - The selected subsystem radio button will be highlighted.
 - The components corresponding to that the subsystem will be displayed in the component window.
- 3 Select a component by clicking its name under the component window.
 - All the servers for the selected component will be displayed in the server window.
- 4 If desired, click the **monitor** button from the common tasks window.
 - This will invoke the ECS Monitor GUI window as shown in Fig. 6.2.2-1.
 - The status “UP/DOWN” indicates whether the server is running.

To see which host each server is running on, click the **cdsping all servers...** button.

- This will invoke the ECS Monitor (cdsping) GUI as indicated in Fig. 6.2.2-2.
- The host name for each running server is listed

Both ECS monitor GUI and ECS Monitor (cdsping) GUI can be updated by clicking the **update** button in the GUI.

- This will cause the list to update to the current status.

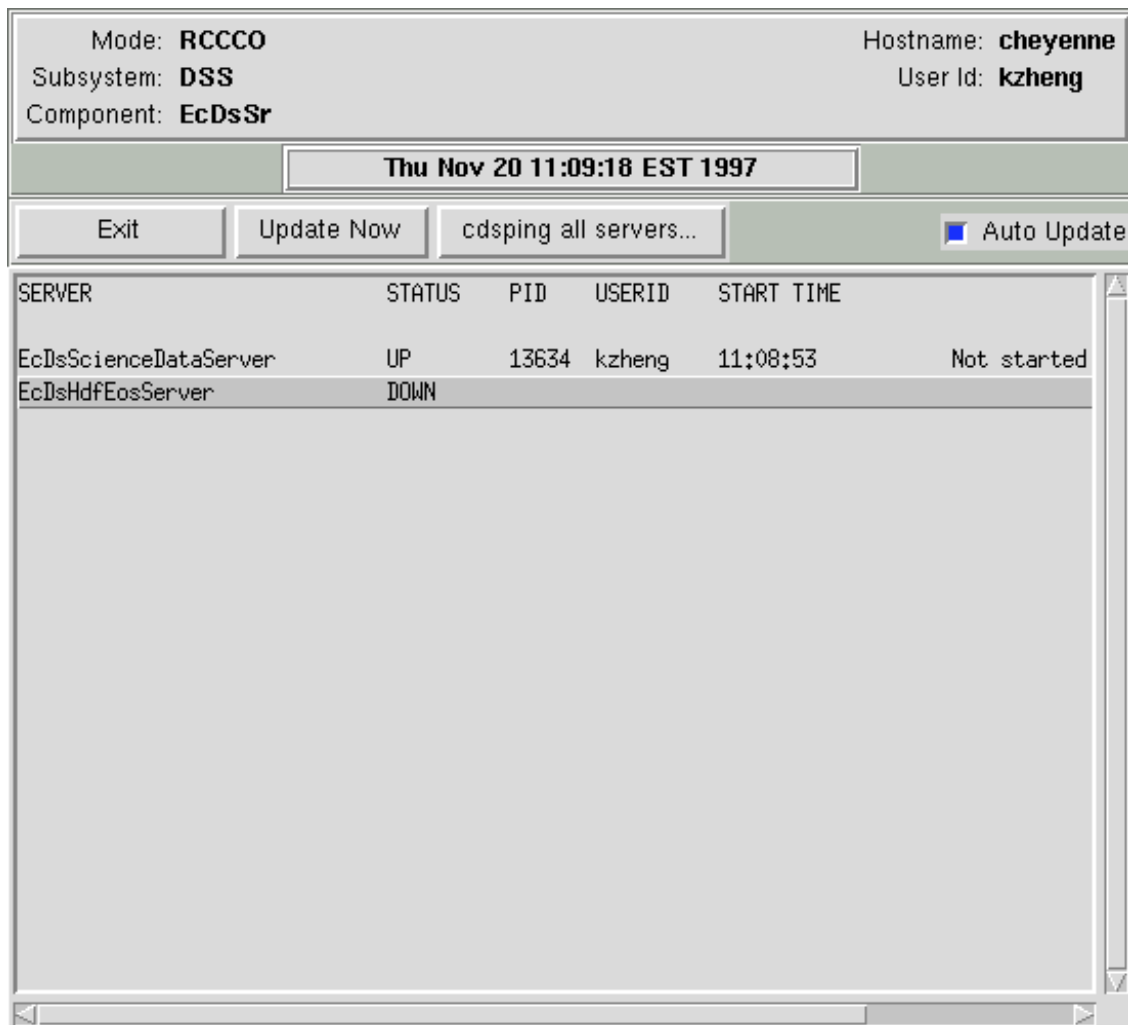


Figure 6.2.2-1. Server Monitor GUI

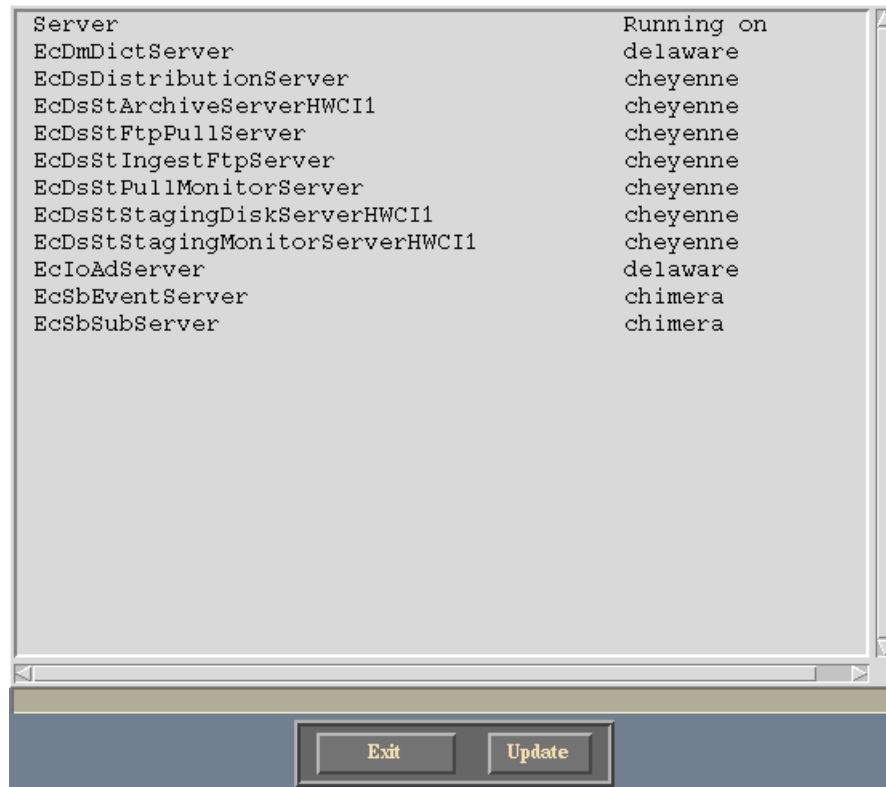


Figure 6.2.2-2. cdsping GUI

- 5 To monitor other servers, repeat steps 2-4.
- 6 To exit, click the **EXIT** button.
 - This will end the monitor GUI.

Table 6.2.2-1. Using ECS Monitor GUI - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Select a mode	Click a mode in the mode list at the ECS Subsystem Manager GUI
2	Select a subsystem	Click the radio button next to the subsystem name under the subsystem component window
3	Select a component	Click a component name under the component window
4	Open the monitor GUI	Click the monitor button from the common tasks window
(4)	See which host each server is running on	Click the cdsping all servers... button.
(4)	Update the ECS monitor GUI and ECS Monitor (cdsping) GUI	Click the update button in the GUIs
5	Monitor other servers	Repeat steps 2-4
6	Exit	Click the EXIT button to end the monitor GUI

6.3 Using ECS Assistant to Manage ESDTs

With the ECS Assistant Tool, ESDTs can be added to and removed from the Science Data Server database by means of a GUI. These operations will be described in the following sections.

6.3.1 Adding ESDTs to the Storage Area

In the ECS system, ESDTs are used to define data granules. Prior to executing a PGE, the relevant ESDT descriptor files and their DLL codes must be installed into the system through the science data server. ECS Assistant provides an easy-to-use GUI to perform these activities. If the installation is successful, the science data server, advertising server, subscription server, Management subsystem (MSS) and the new ESDT implementation library (DLLs) will all be updated.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ECS Assistant has been properly installed.
2. The ECS Assistant GUI is running.

3. Proper clearcase view has been set.
4. The ESDT descriptor files are installed in the specified mode.

To add ESDTs through the ECS Assistant GUI, execute the procedure steps that follow:

- 1 Launch the ECS Assistant GUI as described in **Section 6.1.1**.
 - The GUI will display three selection buttons, one of which is **ESDT Manager**.
- 2 Click the **ESDT Manager** button to open the ESDT Manager GUI.
 - The ESDT Manager GUI will be invoked as shown in Fig. 6.3.1-1.

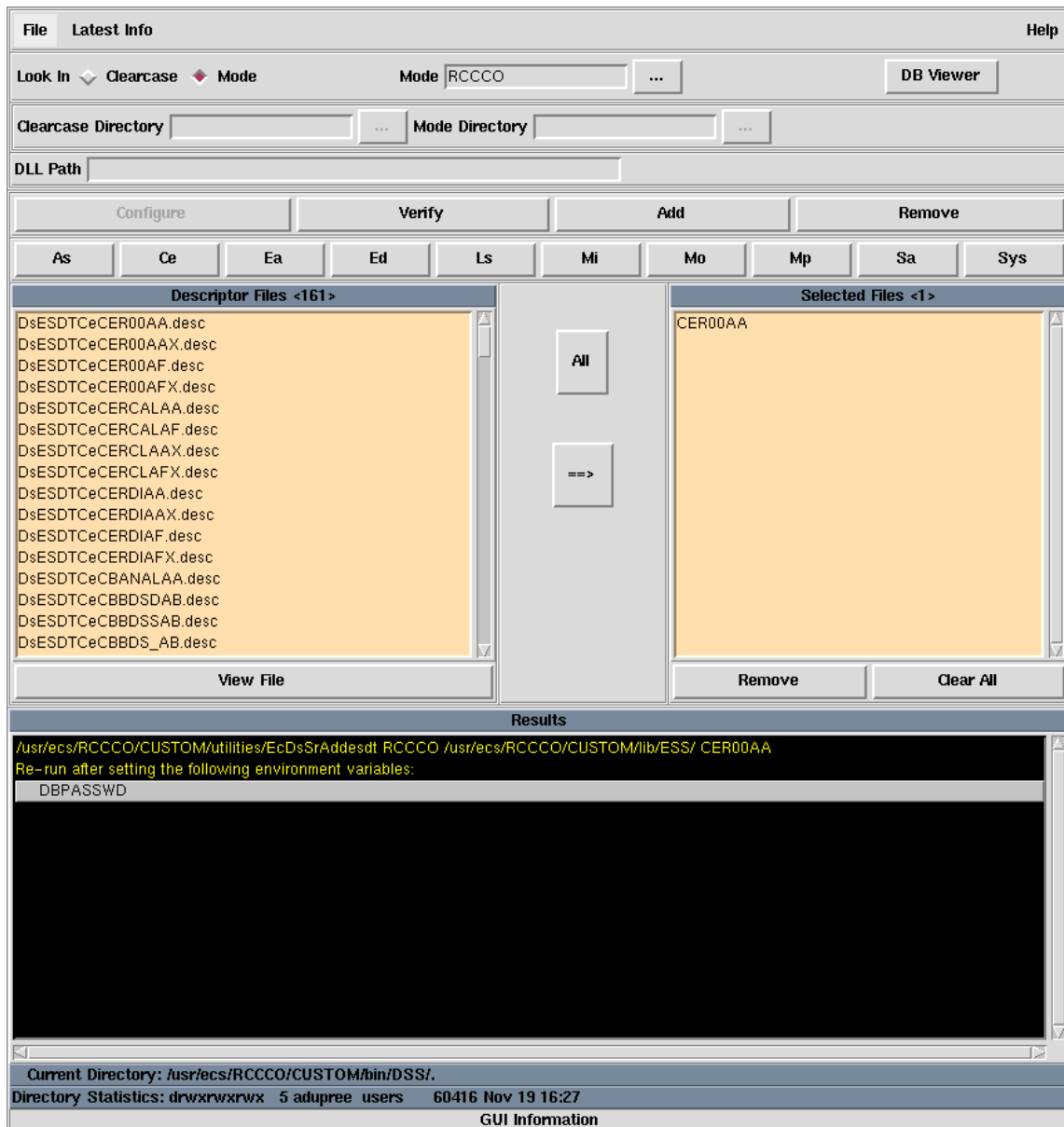


Figure 6.3.1-1. ESDT Manager GUI.

- 3 Select a mode by clicking the drop down list in the **Mode** window.
 - All the modes are listed in the **Mode** window.
 - Select a mode by clicking its name in the mode list.
- 4 Input a DLL path where the shared object files are located by typing a full path name in the **DLL Path** window.
- 5 Select a subdirectory for the instrument team by clicking the corresponding abbreviation.
 - The descriptor files installed for that instrument will be listed.
- 6 Select descriptor files to be added to the archive by clicking the descriptor names in the list.
 - The selected descriptor files are highlighted.
- 7 Move the selected files to the **Selected Files** window by clicking the “==>” button.
 - The short names for the selected descriptor files are listed in the Selected Files window.
- 8 To add the selected files into the archive, click the **Add** button.
 - The ESDTs are added to the archive.
- 9 To exit, select **Exit** in the **File** pull-down menu.
 - This will terminate the ESDT Manager GUI.

Table 6.3.1-1. Running ESDT Add GUI - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Launch the ECS Assistant GUI	Type EcCoAssist & from the /tools/common/ea directory (see Section 6.1.1)
2	Open the ESDT Manager GUI	Click the ESDT Manager button
3	Select a mode	Click the drop down list in the Mode window
4	Input a DLL path where the shared object files are located	Type a full path name in the DLL Path window
5	Select a subdirectory for the instrument	Click the corresponding abbreviation
6	Select descriptor files to be added	Click the descriptor names in the list
7	Move the selected files to the Selected Files window	Click the “==>” button
8	Add the selected files to the archive	Click the Add button
9	Exit	Select File..Exit to dismiss the ESDT Add GUI.

6.3.2 Removing ESDTs via the Science Data Server.

If an ESDT is installed incorrectly or a new version of it becomes available, the installed ESDT will need to be removed from the archive, so that the new ESDT can be added. As with addition of ESDTs, their removal can also be accomplished with the ECS Assistant. Doing so performs the following actions:

- The selected ESDT is removed from DsSrConfiguration.acfg file
- The selected ESDT is removed from the database table
- The event file for the selected ESDT is removed
- The destination descriptor and DLL files for the selected ESDT are removed.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ECS Assistant has been properly installed.
2. The ECS Assistant GUI is running.
3. ESDTs are not stored under Clearcase.
4. The ESDT descriptor files are installed in the specified mode.

To use the ECS Assistant GUI to remove ESDTs from the archive area, execute the procedure steps that follow:

- 1 Open the ECS Assistant GUI as described in **Section 6.1.1**.
 - The GUI will display three selection buttons, one of which is **ESDT Manager**.
- 2 At the ECS Assistant GUI, click the **ESDT Manager** button to open the ESDT Manager GUI.
 - The ESDT Manager GUI will be invoked as shown in Fig. 6.3.1-1.
- 3 Select a mode by clicking the drop down list in the Mode window.
 - All the modes are listed in the Mode window.
 - Select a mode by clicking its name in the mode list.
- 4 Input a DLL path where the shared object files are located by typing a full path name in the DLL Path window.
- 5 Select a subdirectory where the ESDTs are staged by clicking a subdirectory name (as indicated by the instrument team abbreviation).
 - The descriptor files stored in that directory are listed.
- 6 Select descriptor files to be removed from the archive by clicking their names in the list.

- The selected descriptor files are highlighted.
- 7 Move the selected files to the Selected Files window by clicking the “==>” button.
- The short name for the selected descriptor files are listed in the Selected Files window.
- 8 To remove the selected ESDTs from the archive, click the **Remove** button.
- The ESDTs are removed from the archive.
 - The necessary cleanups are performed.
- 9 To exit, click the **EXIT** in the **File** pull down menu.
- This will end the ESDT Manager GUI.

Table 6.3.2-1. Using ECS Assistant to Remove ESDTs - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Open ECS Assistant GUI	As described in Section 6.1.1.
2	Open the ESDT manager GUI.	Click the ESDT Manager button
3	Select a mode	Click the drop down list in the Mode window and select a mode by clicking the mode name in the mode list.
4	Input a DLL path where the shared object files are located	Type a full path name in the DLL Path window
5	Select a subdirectory where the ESDTs are staged	Click a subdirectory name (as indicated by the instrument team abbreviation).
6	Select descriptor files to be removed from the archive	Click the descriptor names in the list.
7	Move the selected files to the Selected Files window	Click the “==>” button.
8	Remove the selected ESDTs from the archive	Click the Add button.
9	Exit	Select File..EXIT

6.3.3 Using ECS Assistant to View ECS Science Data Server Database

ESDTs and their granules stored in the archive are managed using an ECS Science Dataserver database. ECS Assistant provides an easy way to review the records stored in this database by using the ECS Assistant DB Viewer. There are two main windows in the DB Viewer. The first is called Collections and is used to display ESDT information included in the Collection database table. Information listed in this table includes ESDT short names, times last updated, types, etc. If an ESDT is added to the Science Data Server, its record will be shown in this window. The other window is called Granules and is used to display information included in the Granule database table. If a granule is inserted for an ESDT, the granule information will be listed in this window if its ESDT is highlighted in the Collection window. In addition to these two main windows, this DB Viewer GUI can also show ESDT database validation rules, PSA information, and summary information about the database reviewed.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ECS Assistant has been properly installed.
2. The ECS Subsystem Manager is running.
3. The environment variables for using the database have been set correctly.

To start up the ECS monitor GUI, execute the procedure steps that follow:

- 1 Follow Section 6.1.1 to invoke the ECS Assistant GUI.
 - The ECS Assistant GUI will be launched.
- 2 At the ECS Assistant GUI, select ESDT Manager GUI by clicking the ESDT Manager.
 - The ESDT manager GUI will appear.
- 3 At the ECS ESDT Manager GUI, select the DB Viewer by clicking the **DB Viewer** button.
 - The Database Login GUI will appear as shown in Fig. 6.3.3-1.
 - Fill in the fields to point to the specific database for the mode used.
 - Click Login to open the DB Viewer.
 - The DB Viewer GUI will appear as shown in Fig. 6.3.3-2.
 - ESDTs are listed in the Collections window.

DB user:	<input type="text" value="rmagill"/>	\$DSQUERY:	<input type="text" value="relbsgi_sqs222_srvr"/>
Password:	<input type="password" value="*****"/>	\$SYBASE:	<input type="text" value="/tools/sybOCv11.1.0"/>
Database name:	<input type="text" value="relbdss_RCCCO"/>		

Fill in the three fields above, or fill in the Mode below and press Use to obtain the database info from the mode's configuration file. Then press Login. If you started with the mode as a command line argument, everything should be filled in and you can just press Login. If the values for \$SYBASE and \$DSQUERY are not right, you will need to set them in your environment and restart.

Mode:	<input type="text" value="RCCCO"/>	<input type="button" value="Use"/>
-------	------------------------------------	------------------------------------

Figure 6.3.3-1. Database Login GUI

Collections					
Short Name	Last Update		type	subType	Long Name
AEROSAG2	Nov 6 1997	4:39:08:196PM	Science	AEROSAG2	Langley Research Center SAGE II Aerosol
AM1ANC	Jul 16 1997	4:33:38:426PM	Science	AM1ANC	AM-1 Ancillary APIDx4
AM1ATTH	Jul 16 1997	4:29:14:693PM	LIMITED	AM1ATTH	Preprocessed AM-1 Platform Attitude Data in HD
AM1ATTN	Jul 16 1997	4:30:37:696PM	LIMITED	AM1ATTN	Preprocessed AM-1 Platform Attitude Data in Na
AM1EPHMH	Jul 16 1997	4:26:25:660PM	LIMITED	AM1EPHMH	Preprocessed AM-1 Platform Ephemeris Data in H
AM1EPHMN	Jul 16 1997	4:32:04:843PM	LIMITED	AM1EPHMN	Preprocessed AM-1 Platform Ephemeris Data in N
AM1Ephem	Oct 9 1997	9:20:53:200AM	Science	AM1Ephem	Predicted EOS AM-1 Ephemeris
AM1GDTrk	Oct 8 1997	1:34:55:133PM	Science	AM1GDTrk	Predicted Sub-Satellite Point (Ground Track)
AM1OrNum	Oct 8 1997	1:36:26:116PM	Science	AM1OrNum	Predicted Orbit Number and Start Times
AST0SCS	Sep 9 1997	1:23:05:093PM	Science	AST0SCS	SWIR/calibration
AST0SCSE	Sep 9 1997	1:24:21:083PM	Science	AST0SCSE	SWIR/calibration
AST0SS	Sep 10 1997	9:52:39:196AM	Science	AST0SS	SWIR/observation
AST0STSE	Sep 9 1997	1:27:22:110PM	Science	AST0STSE	SWIR/test
AST0TCE	Sep 9 1997	1:28:54:126PM	Science	AST0TCE	TIR/calibration
AST0TCS	Sep 9 1997	1:30:23:143PM	Science	AST0TCS	TIR/calibration
AST0TCSE	Sep 9 1997	1:31:49:113PM	Science	AST0TCSE	TIR/calibration
AST0TE	Sep 9 1997	4:04:24:250PM	Science	AST0TE	TIR/observation
AST0TS	Sep 9 1997	4:06:01:033PM	Science	AST0TS	TIR/observation
AST0TSE	Sep 9 1997	4:07:26:056PM	Science	AST0TSE	TIR/observation
AST0TTE	Sep 9 1997	4:08:52:070PM	Science	AST0TTE	TIR/test
AST0TTS	Sep 9 1997	4:10:18:080PM	Science	AST0TTS	TIR/test
AST0TTSE	Sep 10 1997	8:42:01:126AM	Science	AST0TTSE	TIR/test
AST0V1C	Sep 10 1997	8:45:33:156AM	Science	AST0V1C	VNIR(1)/calibration

Granules					
dbID	insertTime	LocalGranuleID	BeginningDateTime	EndingDateTime	
4195	Oct 16 1997 3:43:47:270PM	LocalGranuleID	Jan 1 1993 1:00AM	Jan 30 1993 2:0	

Figure 6.3.3-2. DB Viewer GUI

- 4 To view the inserted granules for a selected ESDT, first select an ESDT by clicking its short name in the Collections window.
 - The selected ESDT is highlighted.
 - Granule information for that ESDT, if there is any, will be listed in the Granules window.
- 5 To exit, click the **EXIT** button.
 - This will end the DB Viewer GUI.

Table 6.3.3-1. Using ECS Assistant to View ECS Science Data Server Database - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Invoke the ECS Assistant GUI.	Follow Section 6.1.1
2	At the ECS Assistant GUI, invoke the ESDT Manager	Click the ESDT Manager
3	At the ECS ESDT Manager GUI, select the DB Viewer to open the login window for the database	Click the DB Viewer button.
4	Fill in the Login field and open DB Viewer for the selected database	Follow the GUI guidance to fill in the fields and press LOGIN
5	View the inserted granules for a selected ESDT	Select an ESDT by clicking the ESDT short name in the Collections window.
6	Exit	Click the EXIT button to end the DB Viewer GUI.

This page intentionally left blank.

7. The SSIT Manager

The principal tool used during SSI&T is the SSIT Manager. The SSIT Manager is the top-level graphical user interface (GUI) environment presented to SSI&T personnel. Its purpose is to bring together the tools needed for SSI&T into a single, graphical environment.

Across the top of the SSIT Manager are the toolbar items **File**, **Tools**, and **Run**. Clicking on each of these invokes a pull-down menu.

Under the **File** pull-down menu, the only item is **Exit**. Clicking on this causes the SSIT Manager to terminate.

The **Tools** pull-down menu has most of the SSIT Manager's tools. The menu items are:

- **Code Analysis** contains
 - **SPARCwork** - A COTS package provided by Sun that allows for various coding activities including memory checking and debugging.
- **Office Automation** contains
 - **MS Windows** - a Microsoft Windows emulator with MS Office (Word, Excel, PowerPoint) installed.
 - **Ghostview** - for viewing PostScript formatted documents.
 - **Netscape** - WWW browser and useful for viewing HTML formatted documents.
 - **Acrobat** - for viewing PDF formatted documented.
 - **DDTS** - for entering and tracking science software problems.
- **Standards Checkers** contains
 - **FORCHECK** - for standards checking for FORTRAN 77 and Fortran 90 science software source code.
 - **Prohibited Function Checker** - for checking science software source code for prohibited functions.
 - **Process Control File Checker** - for checking Process Control Files (PCFs) delivered with science software.
 - **Prolog Extractor** - for extracting prologs from science software source code.
- **Product Examination** contains

- **IDL** - Interactive Data Language tool.
- **EOSView** - for viewing HDF and HDF-EOS files.
- **File Comparison** contains
 - **ASCII** - for comparing two output products that are in ASCII format.
 - **Binary** - for comparing two output products that are in binary format.
 - **HDF (GUI)** - for comparing two output products that are in HDF or HDF-EOS format, GUI version.
 - **HDF (hdiff)** - for comparing two output products that are in HDF or HDF-EOS format, command line tool.
- **Text Editors** contains
 - **Emacs**
 - **Xedit**
- **PDPS Database** contains
 - **PCF ODL Template** - for converting delivered PCFs into ODL during PGE registration.
 - **Check ODL** - for verifying ODL syntax of ODL files.
 - **SSI&T Science Metadata Update** - for updating the PDPS database with PGE information during PGE registration.
 - **SSI&T Opnl Metadata Update** - GUI for updating the PDPS database with PGE information during PGE registration.
 - **Copy SSIT -> Production** - for copying PGE registration database information from SSI&T mode to Production mode.
- **Data Server** contains
 - **Register Subscription** - for registering a subscription.
 - **Acquire DAP** - for acquiring a Delivered Algorithm Package (DAP).
 - **Insert Static** - for inserting a static data file to the Data Server.
 - **Insert Test Dynamic** - for inserting a dynamic test data file to the Data Server.
 - **Insert EXE TAR** - for inserting a Science Software Executable Package (SSEP) to the Data Server.

- **SSAP Editor** - for editing and creating a Science Software Archive Package (SSAP) and inserting it to the Data Server.

The **Run** pull-down menu initially contains no menu items. Its purpose, however, is to allow a place for SSI&T personnel to place their own custom tools and scripts.

7.1 General Set Up and Running of the SSIT Manager

The SSIT Manager requires a configured environment within which to run; it runs only on the AIT Suns. The set up steps described in this section need only be done the first time a SSI&T operator uses the SSIT Manager.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Table 7.1-1. Assumptions and Limitations

Order	Assumption/Limitation
1	The C shell (or a derivative like T shell) is the current command shell.
2	A file named <code>.cshrc</code> exists in the operator's home directory.
3	The operator is knowledgeable in the use of text editors (such as <i>vi</i> or <i>emacs</i>)

To set up the environment for the SSIT Manager, execute the procedure steps that follow. Note that in the following, `EcsCustomSw` is nominally in the directory `/usr/ecs/Rel_A/CUSTOM` on the AIT Sun:

- 1 At a UNIX prompt on an AIT Sun, type **cp /usr/ecs/mode/CUSTOM/data/DPS/DpAt.pcf \$HOME/mySSITpcf**, press **Return**.
 - The *mode* is the ECS mode in which you are operating. This mode should be **TS1**.
 - The *mySSITpcf* is the file name of the private copy of the PCF that the SSI&T operator will use when running the SSIT Manager. The **\$HOME** is the environment variable for the user's home directory. For example, **cp /usr/ecs/TS1/CUSTOM/data/DPS/DpAt.pcf \$HOME/myPCF**, press **Return**.
- 2 At the UNIX prompt on the AIT Sun, type **setenv PGS_PC_INFO_FILE \$HOME/mySSITpcf**, press **Return**.
 - The *mySSITpcf* is the full path name to the private copy of the PCF to be used with the SSIT Manager when you run it (from step 1).
 - It may be useful to add this line to your `.cshrc` (or other start up script) so that it is set every time you login.
- 3 At the UNIX prompt on the AIT Sun, type **cd /usr/ecs/mode/CUSTOM/bin/DPS**, press **Return**.

- The **mode** is the ECS mode in which you are operating. This mode should be **TS1**.
 - For example, **cd /usr/ecs/TS1/CUSTOM/bin/DPS**, press **Return**.
- 4 At the UNIX prompt on the AIT Sun, type **source .buildrc**, press **Return**.
- This sets environment variables and other settings needed for running the SSIT Manager.
- 5 This step is optional. If the X Window **DISPLAY** parameter has not already been defined, then define it now. Do this by typing **setenv DISPLAY machine:0.0**
- The **machine** is the name of the machine or its IP address on which the SSIT Manager is to be displayed and operated. For example, if the machine name is calahans.hitc.com, then enter **setenv DISPLAY 155.157.123.34:0.0** or **setenv DISPLAY calahans.hitc.com:0.0**.
 - If access to the SSIT Manager will not be from the same machine (all or most of the time), this command should not be added to the .cshrc (or other start up script). Instead, the operator will have to run the command from the command line at the UNIX prompt each time *before* the SSIT Manager is started.
- 6 At the UNIX prompt on the AIT Sun, type **EcDpAtMgr configFile ../cfgr/EcDpAtMgr.CFG ecs_mode mode**, press **Return**.
- The **mode** is the ECS mode in which you are operating. This mode should be **TS1**.
 - This invokes the SSIT Manager GUI which should be displayed.
 - The checklist displayed within the GUI will be the default. To customize this checklist, see Section 7.2.

Table 7.1-2. General Set Up and Running of the SSIT Manager - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	cp /usr/ecs/mode/CUSTOM/data/DPS/DpAt.pcf \$HOME/mySSITpcf	press Return
2	setenv PGS_PC_INFO_FILE \$HOME/mySSITpcf	press Return
3	cd /usr/ecs/mode/CUSTOM/bin/DPS	press Return
4	source .buildrc	press Return
5	(Optional) setenv DISPLAY machine:0.0	press Return
6	DpAtMgr configFile ../cfgr/DpAtMgr.CFG ecs_mode mode	press Return

7.2 Set Up of a Checklist for the SSIT Manager

The SSIT Manager offers the capability of maintaining user-defined checklist of SSI&T activities. The checklist is presented in the main window of the SSIT Manager. A default

checklist is displayed unless a new checklist is specifically created. This procedure explains how to set up a customized checklist.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Table 7.2-1. Assumptions and Limitations

Order	Assumption/Limitation
1	The SSIT Manager has been properly installed. This can be verified by running the SSIT Manager, the sample (default) checklist should be displayed in the main window of the GUI.
2	A private copy of the Process Control File (PCF) used by the SSIT Manager has been created in the user's home directory (see Section 7.1).

See Section 7.1 for general set up and running of the SSIT Manager. See Appendix B for an example of a complete SSI&T checklist.

To create a user-defined checklist for the SSIT Manager, execute the procedure steps that follow:

- 1 At a UNIX prompt on an AIT Sun, type **xterm &**, press **Return**
 - This command starts a new xterm session on the AIT Sun.
- 2 At the UNIX prompt on the AIT Sun, type **cp /usr/ecs/mode/CUSTOM/data/checklist.sample \$HOME/mychecklist**, press **Return**.
 - The **mode** is the ECS mode in which you are operating. This mode should be **TS1**.
 - The **mychecklist** is the file name for a private copy of the checklist file. It will be this copy that will be edited in the steps which follow.
 - **\$HOME** is the user's home directory. Although the checklist file does not have to be in the user's home directory, the following steps assume that it is.
 - For example, type **cp /usr/ecs/TS1/CUSTOM/data/checklist.sample \$HOME/mychecklist**, press **Return**.
- 3 At a UNIX prompt on the AIT Sun, type **vi \$HOME/mychecklist**, press **Return**.
 - The **mychecklist** is the file name of the checklist created in step 1. This command invokes the **vi** editor and reads in the checklist file from the user's home directory. Alternatively, any text editor may be used such as **emacs**. For example, **emacs \$HOME/mychecklist**, press **Return**.
- 4 In the file, search for the lines of the form **DATABASE=ssitUserPathname/filename** and **CHECKLIST=title**. Edit these lines.

- The *ssitUserPathname* is the full path name to where the database files (used in conjunction with the checklist) will be placed. Typically, this is the user's home directory.
 - The *filename* is the base name for the database files. Two database files will be created in the directory given by *ssitUserPathname* using this base name. They will be *filename.dir* and *filename.pag*. For example, using **DATABASE=/home/jdoe/CERESchecklist** will result in two database files, **CERESchecklist.dir** and **CERESchecklist.pag**, being created in **/home/jdoe/**.
 - The *title* is a user-selected name that will appear in the SSIT Manager GUI above the checklist items in the main window.
- 5 In the file, add the items that will appear in the checklist. Search for lines in the file of the form **ITEM=ChecklistStep**. Each of these lines specifies a procedure or step taken during SSI&T. Edit these lines and/or add lines.
- The *ChecklistStep* is any text string (without quotes). Typically, each *ChecklistStep* is a SSI&T operational procedure or task. The source of these steps may be existing documentation, for example, the SSI&T Agreement between the Instrument Team and the DAAC.
 - Add **ITEM** lines as necessary. Checklist items will appear in the SSIT Manager window in the order entered in this file.
 - There is no limitation to the number of **ITEM** lines that can be entered. However, a large checklist will result in the SSIT Manager taking a significantly longer time to start up.
- 6 Save the changes made to the checklist file and exit the editor.
- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.
 - For other editors, refer to that editor's documentation.
- 7 If database files already exist from a previous checklist set up, at the UNIX prompt on the AIT Sun, type **rm ssitUserPathname/filename.dir** press **Return**. Type **rm ssitUserPathname/filename.pag**, press **Return**.
- The *ssitUserPathname* and *filename* are the full path name and file name set in the checklist file in step 4.
 - If these database files already exist (from a previous checklist set up), an error would result when the SSIT Manager was started.
- 8 At a UNIX prompt on the AIT Sun, type **vi \$HOME/mySSITpcf**, press **Return**.
- The *mySSITpcf* is the file name of the private copy of the PCF used by the SSIT Manager (refer to Section 7.1, step 1).
- 9 In the file, search for identifier 603 beginning in the first column. The line should be of the form: **603|DpAtMgrLogDatabaseInit|ssitUserPathname/filename**. Edit this line.
- The *ssitUserPathname* and *filename* are the full path name and file name set in the checklist file in step 4.
- 10 Save the changes made to the Process Control File and exit the editor.

- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.
- For other editors, refer to that editor's documentation.
- The set up of the SSIT Manager's checklist is now complete.

Table 7.2-2. Set Up of a Checklist for the SSIT Manager - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	xterm &	press Return
2	cp /usr/ecs/mode/CUSTOM/data/checklist.sample \$HOME/mychecklist	press Return
3	vi \$HOME/mychecklist	press Return
4	DATABASE= <i>ssitUserPathname/filename</i> CHECKLIST= <i>title</i>	edit in checklist file
5	ITEM= <i>ChecklistStep</i>	add/edit in checklist file
6	Save and quit the editor	invoke editor command
7	(If necessary) rm <i>ssitUserPathname/filename.dir</i> rm <i>ssitUserPathname/filename.pag</i>	press Return press Return
8	vi \$HOME/mySSITpcf	press Return
9	603 DpAtMgrLogDatabaseInit <i>ssitUserPathname/filename</i>	edit in PCF
10	:wq	press Return

This page intentionally left blank.

8. DAP Insert

The **DAP** (Delivered Algorithm Package) is the vehicle by which the PGE, source code, supporting files, documentation, etc. are delivered to a DAAC for SSI&T. Typically, the DAP is a compressed TAR file with a file name of form *string.tar.Z* . After initial processing, the DAP is broken apart into its components and those components will be subsequently processed and used based on their intended function.

The **insert** service is used to put the DAP into the Data Server. Once the DAP is in the Data Server, the **acquire** service is used to retrieve it.

8.1 Performing a DAP Insert

The method described here, for performing a DAP insert, is by command-line prompts and responses.

Assumptions:

1. The proper Sun platform, with the proper directories, setup, etc., is logged onto. In the case of the mini-DAAC, **texas** is one such platform.
2. The C shell (or a derivative) is the current command shell.
3. If running the command-line sequence from an xterm window, the **DISPLAY** environmental parameter must point to the display being used.

The DAP insert sequence may be initiated from the command line. This is done either from a pure telnet connection or from an xterm window . To perform a **DAP insert**, execute the procedure steps that follow:

- 1 Start insert sequence by going to proper location and executing runDttest6

- Type **cd /usr/ecs/OPS/CUSTOM/bin/DSS** <RETURN>
- Type **dsses** <RETURN>

2. The prompt and user response are shown below (notes in italics):

Warning: Could not open message catalog "oodce.cat" *Ignore this line*

Enter dataserver UR (i.e. [VTC:DSSDSRV])=> **[MDC:DSSDSRV]**
 <RETURN>

Possible user entry.

User must include the brackets.



3. The next prompt and user response are:

1. Search for data
2. Insert data
3. Acquire data
4. Delete data
5. Exit


Please make selection=> **2** <RETURN>

The user types in "2"

4. The next prompt/response sequence will be:

Executing insert

Enter short name of main data type=> **AST_06T** <RETURN>

Enter whatever is appropriate 

5. The next prompt/response sequence will be:

What do you wish to insert with the main group?

1. datafiles with metadata and subtypes
2. datafiles with metadata only
3. datafiles only
4. metadata only

Enter choice: **2** <RETURN>

6. The next prompt/response sequence will be:

Enter metadata filename=> **AST_06T.tar.met** <RETURN>

Enter the datafilename [press <CR> to quit]: **AST_06T.tar**
<RETURN>

Enter the datafilename [press <CR> to quit]: **<RETURN>**

The user enters the appropriate metadata filename and datafilename.

The user will keep getting prompted for a datafilename until a line with only a Carriage Return is received.

7. The next prompt/response sequence will be:

Select type of associated group to insert

1. add DAP group
2. add QA group
3. add PRODHISTORY group
4. add BROWSE group
5. add UR group
6. EXIT add associated group menu

Enter selection number: **6** <RETURN>

8. If you get a message indicating success, then you did well.
Otherwise, your actions will be dependent on the nature of the response.

Table 8.1-2. Performing a DAP Insert - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	a. cd /usr/ecs/OPS/CUSTOM/bin/DSS b. dsses	press Return press Return
2	[MDC:DSSDSRV]	press Return
3	2	press Return
4	<i>shortname</i>	press Return
5	2	press Return
6	a. <i>metadata filename</i> b. <i>data filename</i>	press Return
7	6	press Return

This page intentionally left blank.

9. DAP Acquire

The **DAP** (Delivered Algorithm Package) is the vehicle by which the PGE, source code, supporting files, documentation, etc. are delivered to a DAAC for SSI&T. Typically, the DAP is a compressed TAR file with a file name of form *string.tar.Z* . After initial processing, the DAP is broken apart into its components and those components will be subsequently processed and used based on their intended function.

The **insert** service is used to put the DAP into the Data Server. Once the DAP is in the Data Server, the **acquire** service is used to retrieve it.

DAP is acquired from Data Server and placed in the specified directory. Note there will be 2 files, the DAP itself (a big tar file) and the metadata associated with the DAP. The metadata may be helpful in the creating the SSAP.

9.1 Performing a DAP Acquire Using SSIT Manager

Generally, the preferred approach to accomplishing a DAP **acquire** will be through the use of the SSIT Manager GUI.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The following servers/services are up and operational:
Data Server, Subscription Server, Storage management
2. The following has occurred:
 - Ingest has ingested the DAP and **inserted** it into the PDPS database
 - Server has received notification from the Data Server of the **insert**
 - Subscription Server has sent email to the SSIT operator providing notification of DAP **insertion** and the UR of the DAP
3. The SSIT Manager is available

The X Window **DISPLAY** environment variable is pointing to your screen

To perform a DAP **acquire**, execute the procedure steps that follow:

1. If not already on an AIT Sun, log onto one from your current machine.
2. Bring up the SSIT Manager GUI. At the UNIX prompt, type **mgr** (or whatever works!)
3. After a short while, the SSIT Manager GUI will appear. From the SSIT Manager top menu bar, select **Tools -> Data Server -> Acquire DAP**

See figure 9.1-1 . If the SSIT Manager GUI is used to initiate the DAP processing, Step 4 can be skipped.

4. Alternately, one can initiate the DPA processing sequence from the command line. To do this

- Type **source /usr/ecs/TS1/CUSTOM/bin/DPS/.buildrc <RETURN>**
Note: This step only needs to be done once per login
- Type **/usr/ecs/TS1/CUSTOM/bin/DPS/DpAtStageAlgorithmPackage.sh <RETURN>**

5. The user will be prompted with

**** DAP Staging Tool ****

Configuration filename? (enter for default: DpAtAA.CFG)

To respond, type **<RETURN>**

6. The user will be prompted with

ECS Mode of operations? (enter for default: OPS)

To respond, type **TS1 <RETURN>**

7. The user will be prompted with

Name of email message file (including path)?

To respond, type the required file name plus the path, e.g.,
/home/diascone/emessage01.asc <RETURN>

8. The user will be prompted with

Directory to receive staged file?

To respond, type the required directory, e.g.,
/home/diascone/staged <RETURN>

Table 9.1-1. Performing a DAP Acquire - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	(No entry)	log onto AIT Sun
2	mgr (or whatever is appropriate)	press Return
3	select Tools -> Data Server -> Acquire DAP	mouse action
4	a. source /usr/ecs/TS1/CUSTOM/bin/DPS/.buildrc b. /usr/ecs/TS1/CUSTOM/bin/DPS/DpAtStageAlgorithmPackage.sh	press Return press Return
5	(No entry)	press Return
6	TS1	press Return
7	full_path/Email_message_name	press Return
8	staged_file_directory	press Return

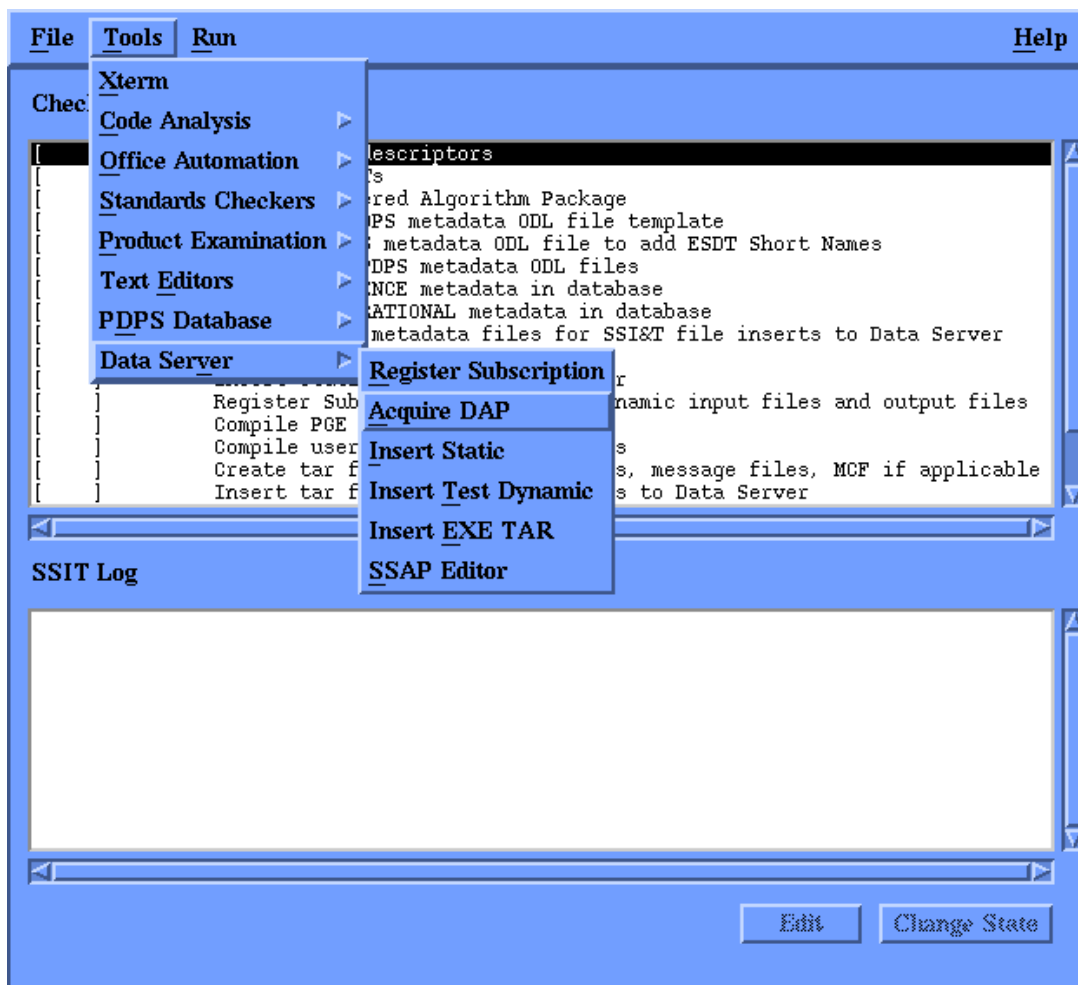


Figure 9.1-1. Selecting DAP Acquire from SSIT Manager

This page intentionally left blank.

10. PGE Checkout

All science software delivered to the DAACs to be run in the ECS must comply with the NASA Earth Science Data and Information Systems (ESDIS) Project standards. The Project standards and guidelines are contained in the document *Data Production Software and Science Computing Facility (SCF) Standards and Guidelines, Revision A, October 1996 (423-16-01)???*.

The currently standards mandate that all science software be written in C, FORTRAN 77, Fortran 90, or Ada. Allowed script languages include C shell, Bourne shell, Korn shell, and Perl. The standards for C, FORTRAN 77, and Fortran 90 are in general ANSI compliance. Certain extensions, particularly in FORTRAN 77, are allowed by the guidelines.

In addition, the ESDIS standards and guidelines document mandates that certain functions are prohibited from science software. These are generally functions (or script utilities) that are a problem for the ECS.

10.1 Checking for ESDIS Standards Compliance in FORTRAN 77

This procedure describes how to use the COTS tool FORCHECK to check science software written in FORTRAN 77 for ESDIS standards compliance.

FORCHECK can be accessed from the UNIX prompt using command-line operations or from the SSIT Manager. The SSIT Manager is a GUI based interface and isolates the user from all of the time consuming steps necessary to set up and run the FORCHECK software. Therefore, it is generally the preferred method to use and is the one described.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The FORTRAN 77 science software source code is available, accessible, and has read permissions for the user.
2. SSIT Manager is available for use.

FORCHECK is available only on the AIT Suns.

To check for ESDIS standards compliance in FORTRAN 77 code, execute the procedure steps that follow:

(Note: Currently, FORCHECK is not available on mini-DAAC)

- 1 If not already on an AIT Sun, log into one from your machine.
 - If necessary, before logging onto AIT Sun, use the xhost command to allow X Window reception.

- Once logged onto proper Sun, remember to set the DISPLAY environmental variable to point to your X Window screen.
- 2 If required, at the UNIX prompt on the AIT Sun, type **cleartool setview *ViewName***, press **Return**.
 - The ***ViewName*** is the name of a view allowing the FORTRAN 77 source files to be accessible.
 - This step is only necessary if any of the FORTRAN 77 source files are in ClearCase (in the VOB under configuration management).
 - 3 If your general environment setup doesn't include transparent access to the SSIT Manager GUI, then you need to set that up. One way to do it is as follows:
 - Set up an alias, manually or from shell script, to set up preliminary environment. At UNIX prompt, type **alias do_buildrc "source /usr/ecs/TS1/CUSTOM/bin/DPS/.buildrc"**
 - Set up an alias, manually or through shell script, to invoke SSIT Manager. At UNIX prompt, type **alias do_ssit_man "/usr/ecs/TS1/CUSTOM/bin/DPS/DpAtMgr ConfigFile /usr/ecs/TS1/CUSTOM/cfg/DpAtMG.CFG ecs_mode TS1& "**
 - 4 Set up the preliminary environment (do_buildrc). This only needs to be done once per session. Then, run SSIT Manager (do_ssit_man).
 - Type **do_buildrc**
 - Type **do_ssit_man**
 - 5 Once the SSIT Manager comes up, the following steps need to be taken to invoke FORCHECK
 - From the top menu bar, select **Tools**.
 - From the Tools menu, select **Standards Checkers**.
 - From the Standards Checkers menu, select **FORCHECK**.
 - See Figure 10.1 for a screen snapshot of this step.
 - 6 A separate FORCHECK window will now open.
 - The user will be prompted for input. The first prompt will be *global option(s) and list file?*
 - The second prompt will be *local option(s) and file(s)?*
 - The second prompt will be repeated until there is a blank line and carriage return.
 - In order to understand what the proper responses should be, the user is encouraged to find hardcopy documentation for FORCHECK or to use the UNIX man facility and type *man forchk* .
 - 7 At the UNIX prompt on the AIT Sun, type **vi FORCHECKoutput**, press **Return**.

- The **FORCHECKoutput** is the file name for the output file produced in step 6.
- The **FORCHECKoutput** file will contain any warnings, errors, and other messages from FORCHECK. A summary will be at the bottom of the file.
- Any text editor may be used for this procedure step.

8 At the UNIX prompt on the AIT Sun, type **vi ListFile**, press **Return**.

- The **ListFile** is the file name for the list file specified at the the FORCHECK prompt.
- The **ListFile** file will contain FORCHECK messages similar to the **FORCHECKoutput** file embedded in the source code listing.
- Any text editor may be used for this procedure step.

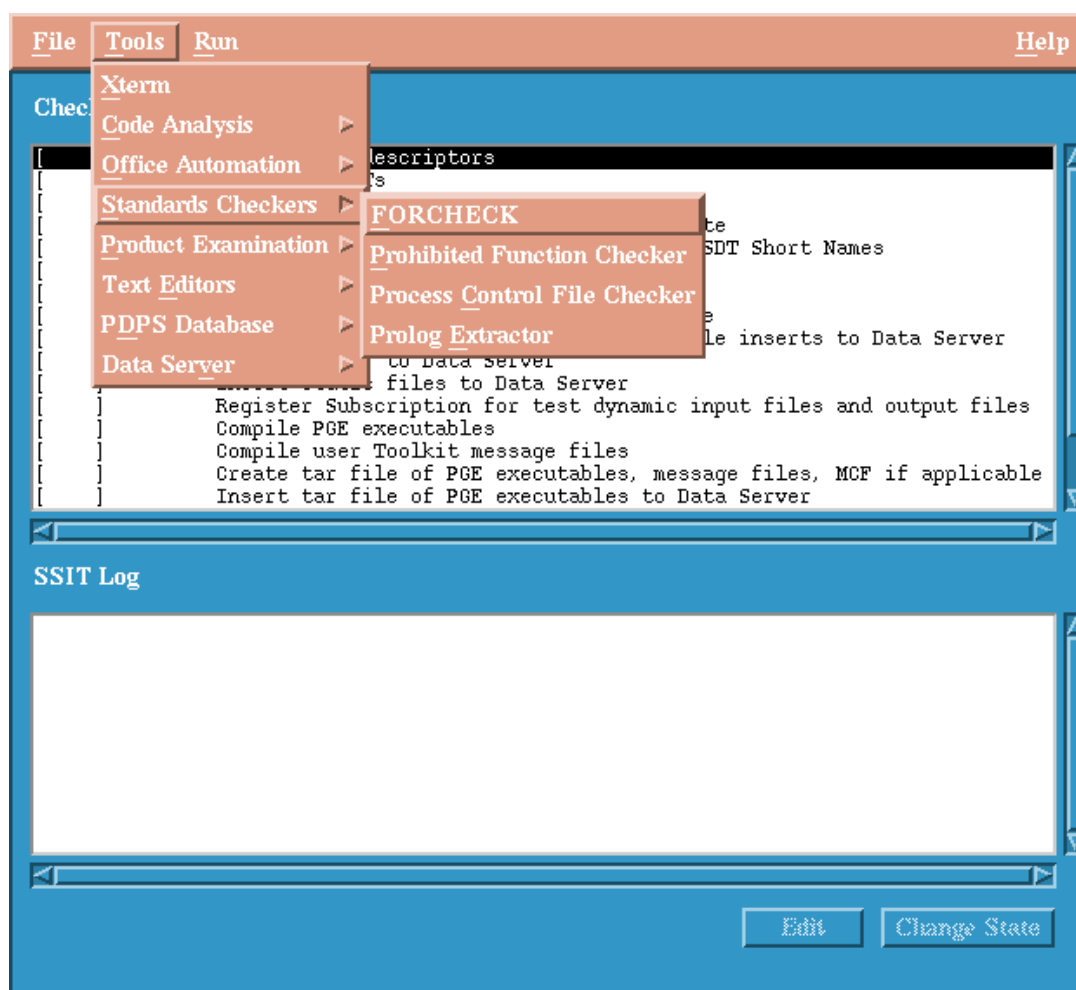


Figure 10.1-1 Screen Snapshot: Accessing FORCHECK from SSIT Manager

Table 10.1-1. Checking for ESDIS Standards Compliance in FORTRAN 77 - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	(No entry)	log onto AIT Sun
2	(Optional) cleartool setview <i>ViewName</i>	press Return
3a	[Optional] alias do_buildrc "source /usr/ecs/TS1/CUSTOM/bin/DPS/.buildrc	press Return
3b	[Optional] alias do_ssit_man "/usr/ecs/TS1/CUSTOM/bin/DPS/DpAtMgr ConfigFile /usr/ecs/TS1/CUSTOM/cfg/DpAtMG.CFG ecs_mode TS1&	press Return
4a	[Optional] do_buildrc	press Return
4b	do_ssit_man (or whatever to start SSIT Manager)	press Return
5	From SSIT Manager top menu, select Tools->Standards Checkers->FORCHECK	No entry
6	Respond to FORCHECK prompts	No entry
7	vi <i>FORCHECKoutput</i>	press Return
8	(Optional) vi <i>ListFile</i>	press Return

10.2 Checking for ESDIS Standards Compliance in Fortran 90

This procedure describes how to use the Fortran 90 compiler flags on the SPR SGI machines to check science software written in Fortran 90 for ESDIS standards compliance.

Unlike with FORTRAN 77, no COTS tool is used to check Fortran 90 science software. Instead, this procedure describes how to use the compiler to perform the checking (ESDIS standards for Fortran 90 are ANSI). Since the Fortran 90 compiler is used, the checking for standards compliance can be naturally tied in with building the science software (since this procedure will produce object files suitable for linking). However, in this procedure, the building of the software (compiling *and* linking) is deferred to a later procedure.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The Fortran 90 science software source code is available, accessible, and has read permissions for the user.
2. Required Status Message Facility (SMF) files have been compiled (see Section 9.3).
3. The C shell (or a derivative) is the current command shell.

4. The Fortran 90 compiler is available on the SPR SGI.

To check for ESDIS standards compliance in Fortran 90 code, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **X**term. Then telnet to the SPR SGI.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SPR SGI.
- 2 At the UNIX prompt on the SPR SGI, set up the proper environment for the compiler to be used by typing **source ToolkitPathname /bin/sgiXX/pgs-dev-env.csh** .
 - **ToolkitPathname** is the home directory of the desired SDP Toolkit version (see Section 9.2).
 - The directory **sgiXX** should be replaced with **sgi32** or **sgi64** as appropriate for the specific compiler desired.
 - For example, on the mini-DAAC platform “lasher”, type **source /data3/ecs/TS1/CUSTOM/daac_toolkit_f90/TOOLKIT/bin/sgi64/pgs-dev-env.csh** . This will set up the various environment parameters, such as **PGSHOME**, to enable the 64 bit version of the FORTRAN 90 compiler to be run.
- 3 If required, at the UNIX prompt on the SPR SGI, type **cleartool setview ViewName**, press **Return**.
 - The **ViewName** is the name of a view allowing the Fortran 90 source files to be accessible.
 - This step is only necessary if any of the Fortran 90 source files are in ClearCase (in the VOB under configuration management).
- 4 At the UNIX prompt on the SPR SGI, type **cd SrcPathname**, press **Return**.
 - The **SrcPathname** is the full path name to the location of the Fortran 90 source files to be checked.
 - The **SrcPathname** will be in the ClearCase VOB if the Fortran 90 source files are checked into ClearCase.
- 5 At the UNIX prompt on the SPR SGI, type **f90 -c -ansi [-I\$PGSINC] [-I\$HDFINC] [[-IOtherIncFiles]...] SourceFiles >& ReportFile**, press **Return**.
 - The terms in square brackets (*/* */*) are used to optionally specify locations of include and module (.mod) files. The **\$PGSINC** already contains the SDP Toolkit include directory and **\$HDFINC** already contains the HDF include directory. The **OtherIncFiles** represents one or more additional include or module directories.
 - The **SourceFiles** is a list (space delimited) of Fortran 90 source files or a wildcard template (*e.g.* *.f90).

- The **>&** is a C shell construct that causes standard error (where the output from the Fortran 90 compiler normally emerges) to be redirected to a file.
- The **ReportFile** is the file name under which to save the results of the compile process.
- The **-c** flag causes only compilation (no linking).
- The **-ansi** flag enables ANSI checking.
- Apply the terms in square brackets only as necessary. Do not include the brackets in the actual command. See example below.
- Do not use the **-I** option for include or module files that are in the standard directories or in the current directory.
- The makefile for the science software may contain the names of additional include files needed by the software.
- For example, type **f90 -c -I\$PGSINC -I\$HDFINC -Iecs/modis/pge5/include/*.f90 >& pge10.report**, press **Return**.

6 At the UNIX prompt on the SPR SGI, type **vi ReportFile**, press **Return**.

- The **ReportFile** is the file name for the compilation results as produced in step 5.
- Any text editor may be used for this procedure step.

Table 10.2-1. Checking for ESDIS Standards Compliance in Fortran 90 - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>X</u> term	telnet to SPR SGI
2	source <i>ToolkitPathname</i> /bin/sgiXX/pgs-dev-env.csh	press Return
3	(Optional) cleartool setview <i>ViewName</i>	press Return
4	cd <i>SrcPathname</i>	press Return
5	f90 -c -ansi [-I\$PGSINC] [-I\$HDFINC] [[-IOtherIncFiles]...] <i>SourceFiles</i> >& <i>ReportFile</i>	press Return
6	vi <i>ReportFile</i>	press Return

10.3 Checking for ESDIS Standards Compliance in C

This procedure describes how to use the C compiler flags on the SPR SGI machines to check science software written in C for ESDIS standards compliance.

Unlike with FORTRAN 77, no COTS tool is used to check C science software. Instead, this procedure describes how to use the compiler to perform the checking (ESDIS standards for C are essentially ANSI). Since the C compiler is used, the checking for standards compliance can be naturally tied in with building the science software (since this procedure will produce object files suitable for linking). However, in this procedure, the building of the software (compiling *and* linking) is deferred to a later procedure.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The C science software source code is available, accessible, and has read permissions for the user.
2. Required Status Message Facility (SMF) files have been compiled (see Section 9.3).
3. The C shell (or a derivative) is the current command shell.

The C compiler is available on the SPR SGI.

To check for ESDIS standards compliance in C code, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **Tools** menu, then choose **xterm**. Then telnet to the SPR SGI.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SPR SGI.
- 2 At the UNIX prompt on the SPR SGI, type **setenv PGSHOME ToolkitPathname**, press **Return**. Then type, source **\$PGSHOME/bin/sgi/pgs-dev-env.csh**, press **Return**.
 - The **ToolkitPathname** is the home directory of the desired SDP Toolkit version (see Section 9.2).
 - The **sgiX** refers to the appropriate processor (see Section 9.2). For example, type **source \$PGSHOME/bin/sgi/pgs-dev-env.csh**, press **Return**.
- 3 If required, at the UNIX prompt on the SPR SGI, type **cleartool setview ViewName**, press **Return**.
 - The **ViewName** is the name of a view allowing the C source files to be accessible.
 - This step is only necessary if any of the C source files are in ClearCase (in the VOB under configuration management).
- 4 At the UNIX prompt on the SPR SGI, type **cd SrcPathname**, press **Return**.
 - The **SrcPathname** is the full path name to the location of the C source files to be checked.
 - The **SrcPathname** will be in the ClearCase VOB is the C source files are checked into ClearCase.
- 5 At the UNIX prompt on the SPR SGI, type **cc -c -ansiposix [-I\$PGSINC]/[-I\$HDFINC]/[-IOtherIncFiles]... SourceFiles >& ReportFile**, press **Return**.
 - The terms in square brackets (*[]*) are used to optionally specify locations of include and module (.mod) files. The **\$PGSINC** already contains the SDP Toolkit include

directory and **\$HDFINC** already contains the HDF include directory. The **OtherIncFiles** represents one or more additional include directories.

- The **SourceFiles** is a list (space delimited) of C source files or a wildcard template (*e.g.* *.c).
- The **>&** is a C shell construct that causes standard error (where the output from the C compiler normally emerges) to be redirected to a file.
- The **ReportFile** is the file name under which to save the results of the compile process.
- The **-c** flag causes only compilation (no linking).
- The **-ansiposix** flag enables ANSI and POSIX checking.
- Apply the terms in square brackets only as necessary. Do not include the brackets in the actual command. See example below.
- Do not use the **-I** option for include files that are in the standard directories (*e.g.* /usr/include) or in the current directory.
- The makefile for the science software may contain the names of additional include files needed by the software.
- For example, type **cc -c -I\$PGSINC -I\$HDFINC -Iecs/modis/pge5/include/ *.c >& pge10.report**, press **Return**.

6 At the UNIX prompt on the SPR SGI, type **vi ReportFile**, press **Return**.

- The **ReportFile** is the file name for the compilation results as produced in step 5.
- Any text editor may be used for this procedure step.

Table 10.3-1. Checking for ESDIS Standards Compliance in C - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Tools → Xterm	telnet to SPR SGI
2	setenv PGSHOME <i>ToolkitPathname</i> source \$PGSHOME/bin/sgiX/pgs-dev-env.csh	press Return press Return
3	(Optional) cleartool setview <i>ViewName</i>	press Return
4	cd <i>SrcPathname</i>	press Return
5	cc -c -ansiposix [-I\$PGSINC] [-I\$HDFINC] [[-IOtherIncFiles]...] <i>SourceFiles</i> >& <i>ReportFile</i>	press Return
6	vi <i>ReportFile</i>	press Return

10.4 Checking for ESDIS Standards Compliance in Ada

(Not available on mini-DAAC)

This procedures describes how to use Ada compilers on the SPR SGI machines to check science software written in Ada for ESDIS standards compliance.

Unlike with FORTRAN 77, Fortran 90, or C, Ada compilers are subjected to a validation process by the DoD Ada Committee. Thus, any code that compiles successfully by a validated compiler is, by definition, fully ANSI compliant. Since the Ada compiler is used, the checking for standards compliance can be naturally tied in with building the science software (since this procedure will produce object files suitable for linking). However, in this procedure, the building of the software (compiling *and* linking) is deferred to a later procedure.

Section 10.4.1 describes how to use the COTS Verdix compiler and 8.4.2 describes how to use the GNU *gcc* compiler.

10.4.1 Checking for ESDIS Standards Compliance in Ada: Verdix COTS

This procedure describes compiling Ada software using the COTS Verdix Ada Development System (VADS) which provides a complete environment for building (and developing) Ada software. See Section 8.4.2 for use of the *gcc* compiler in compiling Ada code.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The Ada science software source code is available, accessible, and has read permissions for the user.
2. The C shell (or a derivative) is the current command shell.

The Ada compiler is available on the SPR SGI.

To check for ESDIS standards compliance in Ada code, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then telnet to the SPR SGI.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SPR SGI.
- 2 If required, at the UNIX prompt on the SPR SGI, type **cleartool setview *ViewName***, press **Return**.
 - The ***ViewName*** is the name of a view allowing the Ada source files to be accessible.
 - This step is only necessary if any of the Ada source files are in ClearCase (in the VOB under configuration management).
- 3 At the UNIX prompt on the SPR SGI, type **setenv SGI_ABI -32**, press **Return**.
 - This command sets the environment variable **SGI_ABI** for 32-bit mode compilation.

- 4 At the UNIX prompt on the SPR SGI, type **cd *SrcPathname***, press **Return**.
 - The *SrcPathname* is the full path name to the location of the Ada source files to be checked.
 - The *SrcPathname* will be in the ClearCase VOB if the Ada source files are checked into ClearCase.
- 5 At the UNIX prompt on the SPR SGI, type **a.mklib**, press **Return**.
 - This command creates a VADS library directory. All Ada compilation must occur in a VADS Ada library.
- 6 At the UNIX prompt on the SPR SGI, type **a.make -v -f *SourceFiles* >& *ReportFile***, press **Return**.
 - The *SourceFiles* is a list (space delimited) of Ada source files or a wildcard template (e.g. *.ada).
 - The >& is a C shell construct that causes standard error (where the output from the Ada compiler normally emerges) to be redirected to a file.
 - The *ReportFile* is the file name under which to save the results of the compile process.
 - The -v flag enables verbose output.
 - The -f flag indicates that what immediately follows are the source files. The order of the flags is therefore important.
- 7 At the UNIX prompt on the AIT Sun, type **vi *ReportFile***, press **Return**.
 - The *ReportFile* is the file name for the compilation results as produced in step 6.
 - Any text editor may be used for this procedure step.

Table 10.4.1-1. Checking for ESDIS Standards Compliance in Ada: Verdix COTS - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>X</u> term	telnet to SPR SGI
2	(Optional) cleartool setview <i>ViewName</i>	press Return
3	setenv SGI_ABI -32	press Return
4	cd <i>SrcPathname</i>	press Return
5	a.mklib	press Return
6	a.make -v -f <i>SourceFiles</i> >& <i>ReportFile</i>	press Return
7	vi <i>ReportFile</i>	press Return

10.4.2 Checking for ESDIS Standards Compliance in Ada: GNU *gcc* Compiler

(Not available on mini-DAAC)

This procedure describes compiling Ada software using the GNU C compiler, *gcc*. See Section 8.4.1 for use of the COTS Verdex compiler in compiling Ada code.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The Ada science software source code is available, accessible, and has read permissions for the user.
2. The C shell (or a derivative) is the current command shell.

The GNU *gcc* compiler is available on the SPR SGI.

To check for ESDIS standards compliance in Ada code, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **X**term. Then telnet to the SPR SGI.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SPR SGI.
 - It is recommended that this procedure begin within a new command shell on the SPR SGI.
- 2 If required, at the UNIX prompt on the SPR SGI, type **cleartool setview *ViewName***, press **Return**.
 - The ***ViewName*** is the name of a view allowing the Ada source files to be accessible.
 - This step is only necessary if any of the Ada source files are in ClearCase (in the VOB under configuration management).
- 3 At the UNIX prompt on the SPR SGI, type **setenv SGI_ABI -32**, press **Return**.
 - This command sets the environment variable **SGI_ABI** for 32-bit mode compilation.
- 4 At the UNIX prompt on the SPR SGI, type **cd *SrcPathname***, press **Return**.
 - The ***SrcPathname*** is the full path name to the location of the Ada source files to be checked.
 - The ***SrcPathname*** will be in the ClearCase VOB is the Ada source files are checked into ClearCase.

- 5 At the UNIX prompt on the SPR SGI, type **gcc -c -gnat83 SourceFiles >& ReportFile**, press **Return**.
 - The *SourceFiles* is a list (space delimited) of Ada source files or a wildcard template (e.g. *.ada).
 - The >& is a C shell construct that causes standard error (where the output from the gcc compiler normally emerges) to be redirected to a file.
 - The *ReportFile* is the file name under which to save the results of the compile process.
 - The -c flag causes only compilation (no linking).
 - The -gnat83 enables compilation of Ada using the 1983 Ada Standard. Note that without this flag, the compiler would assume the 1995 Ada proposed Standard.
- 6 At the UNIX prompt on the SPR SGI, type **vi ReportFile**, press **Return**.
 - The *ReportFile* is the file name for the compilation results as produced in step 5.
 - Any text editor may be used for this procedure step

**Table 10.4.2-1. Checking for ESDIS tandards Compliance in Ada:
GNU gcc Compiler - Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>X</u> term	telnet to SPR SGI
2	(Optional) cleartool setview <i>ViewName</i>	press Return
3	setenv SGI_ABI -32	press Return
4	cd <i>SrcPathname</i>	press Return
5	gcc -c -gnat83 <i>SourceFiles</i> >& <i>ReportFile</i>	press Return
6	vi <i>ReportFile</i>	press Return

10.5 Checking for Prohibited Functions

The Project standards and guidelines are contained in the document *Data Production Software and Science Computing Facility (SCF) Standards and Guidelines, Revision A, October 1996* (423-16-01). This ESDIS document mandates that science software delivered to the DAACs to be integrated into the ECS be free of prohibited functions. The procedures that follow describe how to use the Prohibited Function Checker, available with a GUI and without.

Section 10.5.1 describes how to use the Prohibited Function Checker GUI. Section 10.5.2 describes how to use the Prohibited Function Checker from the command line. Both versions of the checker are functionally identical.

10.5.1 Checking for Prohibited Functions: GUI Version

This procedure describes using the GUI version of the Prohibited Function Checker to check science software for the prohibited functions.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The SSIT Manager is running and that the source files to be checked are available, accessible, and have read permissions for the operator.
2. Source files to be checked are Ada, C, FORTRAN 77, Fortran 90, C shell, Korn shell, Bourne shell, or Perl and have recognizable file name extensions (refer to Table 8.5.1-2).

Table 10.5.1-1. File Name Extensions Recognized

Language	File Name Extensions
Ada	.a, .ada
C	.c, .h
FORTRAN 77	.f, .f77, .ftn
Fortran 90	.f90
C Shell	.csh
Korn Shell	.ksh
Bourne Shell	.sh
Perl	.pl

To check for prohibited functions in delivered source files, execute the procedure steps that follow:

Assumptions:

1. The SSIT Manager is running.
- 1 From the SSIT Manager, click on the **Tools** menu, then choose **Standards Checkers**. Then choose **Prohibited Function Checker**. See Figure 10.5.1-1 .
 - The Prohibited Function Checker GUI will be displayed.
 - See Figure 10.5.1-2 .
- 2 In the Prohibited Function Checker GUI, click on the **Analyze** button.
 - The File Selector GUI will be displayed.
 - See Figure 10.5.1.3 .
- 3 Within the **Directories** subwindow, work your way to the desired directory.
 - When you have reached the directory you want, the files for that directory will be displayed in the **Files:** subwindow.

- 4 Within the **Files** subwindow, click on the source files to be checked. Each file clicked on will be highlighted.
 - To choose groups of contiguous files, hold down the left mouse button and drag the mouse.
 - To choose groups of contiguous files with a single button mouse (e.g., Macintosh), hold down the mouse button and shift key.
 - To choose non-contiguous files, hold down the Control key while clicking on file names.
 - See Figure 10.5.1.4 for an example of the **Files:** subwindow after several files have been selected and consequently highlighted,
- 5 In the File Selector GUI, click on the **Ok** button.
 - The File Selector GUI will disappear.
 - The files selected in step 5 will be displayed in the **Prohibited Function Checker** GUI window as they are being checked.
 - If no prohibited functions are found, the **Prohibited Function Checker** GUI will be displayed with the message **no prohibited functions found** . You can then go to step 9 (to quit) or go to step 2 and pick more files to analyze.
 - If prohibited functions were found, proceed to step 6.
- 6 In the Prohibited Function Checker GUI, click on the **Report** button.
 - The **Report** GUI will be displayed.
 - For each file, a list of prohibited functions found will be displayed.
- 7 Optionally, click on the **Print** button or the **Save** button.
 - Choose **Save** to save the results to a file; choose **Print** to have the results printed on the default printer.
 - Choosing **Save** will bring up a GUI labeled **Save To File**. Specify the directory and file name in which to save the results file.
- 8 Optionally, in the Prohibited Function Checker GUI, highlight one of the source files listed. Then click on **View**.
 - The **Source Code** GUI will be displayed.
 - Occurrences of prohibited functions found in that source file will be highlighted.
 - Click on the **Next** button to bring into the window successive occurrences of prohibited functions (the **Next** button does not bring in the next source file).
 - Click on the **Done** button to close the **Source Code** GUI. Other source files may be examined similarly, one at a time.
- 9 In the Prohibited Function Checker GUI, click on the **Quit** button.
 - The Prohibited Function Checker GUI will disappear.

- This ends the session.

Table 10.5.1-2. Checking for Prohibited Functions: GUI Version - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Tools → Standards Checkers → Prohibited Function Checker	(No action)
2	Analyze button	click button
3	directory	click name
4	file(s)	click name(s)
5	Ok	click button
6	(Optional) Report	click button
7	(Optional) Save Print	click button
8	(Optional) View	click button

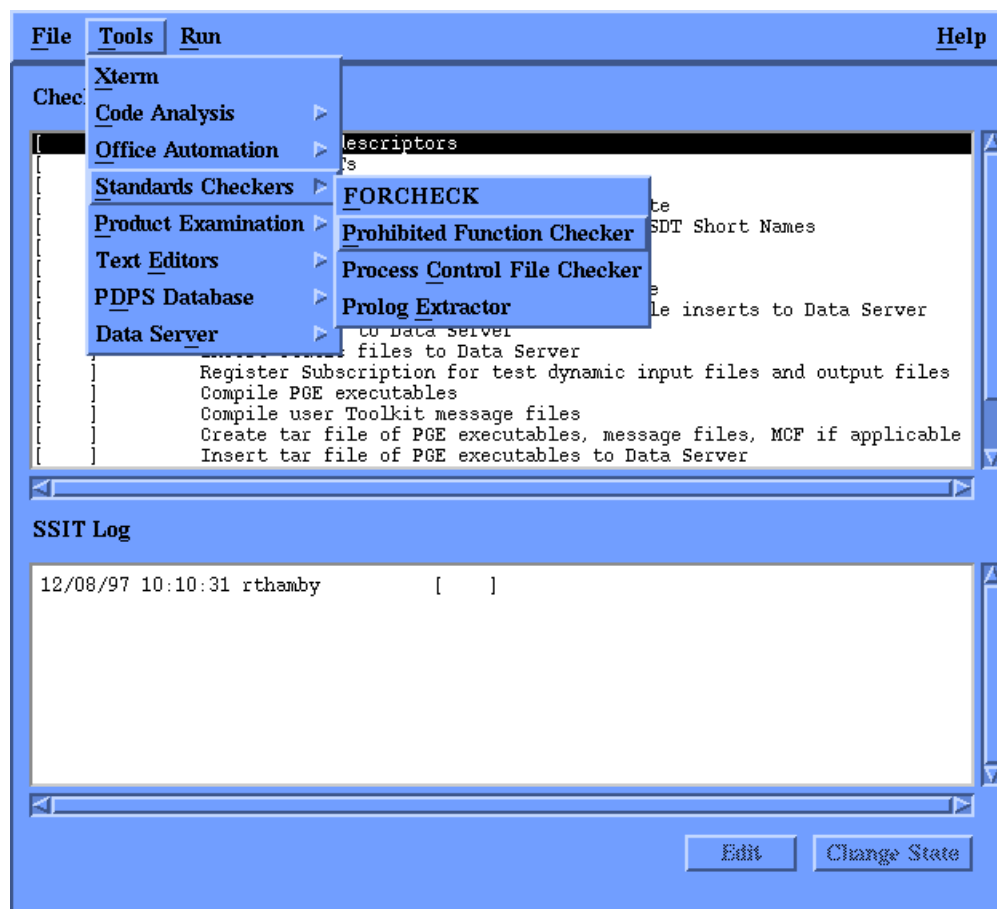


Figure 10.5.1-1. Invoking the Prohibited Function Checker

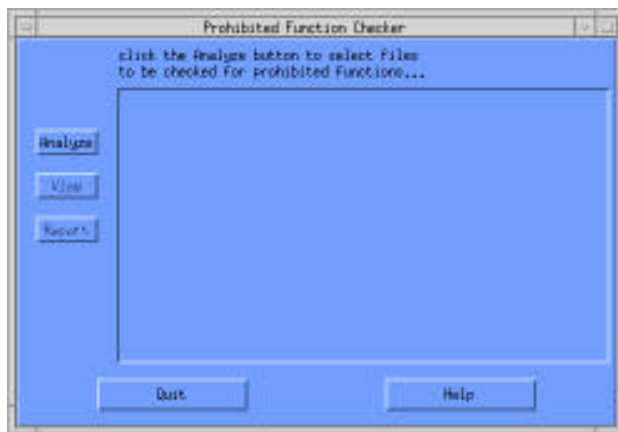


Figure 10.5.1-2. Starting Screen for Prohibited Function Checker

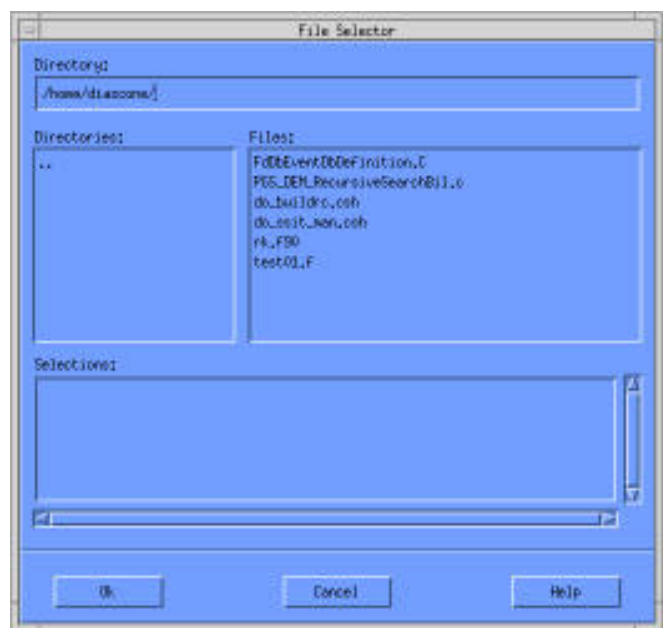


Figure 10.5.1-3. File Selection Menu

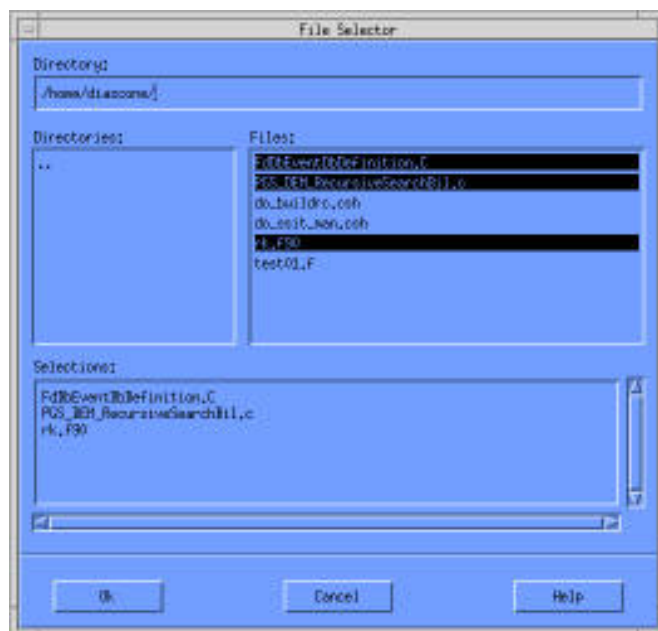


Figure 10.5.1-4. Selected Files

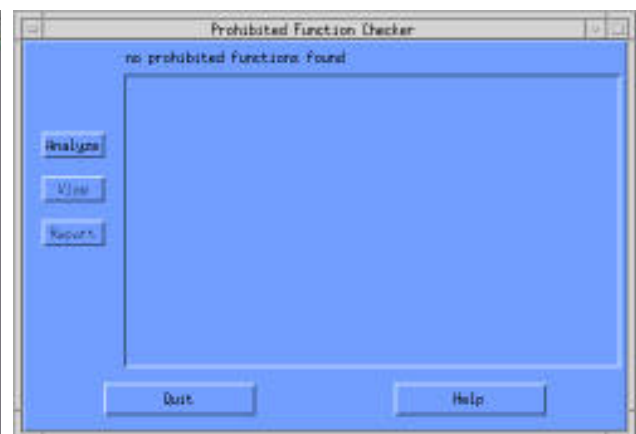


Figure 10.5.1-5. Results Screen

10.5.2 Checking for Prohibited Functions: Command-Line Version

This procedure describes using the command-line version of the Prohibited Function Checker to check science software for the prohibited functions.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The SSIT Manager and its various tools have been installed in the standard directories.
2. The source files to be checked are available, accessible, and have read permissions for the operator.
3. Source files to be checked are Ada, C, FORTRAN 77, Fortran 90, C shell, Korn shell, Bourne shell, or Perl and have recognizable file name extensions (refer to Table 8.5.1-2).

To check for prohibited functions in delivered source files, execute the procedure steps that follow:

- 1 If required, at the UNIX prompt on an AIT Sun, type **cleartool setview *ViewName***, press **Return**.
 - The ***ViewName*** is the name of a view allowing the source files to be accessible.
 - This step is only necessary if any of the source files are in ClearCase (in the VOB under configuration management).
- 2 At the UNIX prompt on the AIT Sun, type **cd *SrcPathname***, press **Return**.
 - The ***SrcPathname*** is the full path name to the location of the source files to be checked.
 - The ***SrcPathname*** will be in the ClearCase VOB if the source files are checked into ClearCase.
 - The ***SrcPathname*** can contain other directories that contain source files and/or more directories. The Prohibited Function Checker will search out all source files in subdirectories recursively.
- 3 At the UNIX prompt on the AIT Sun, type
/data2/ecs/TS1/CUSTOM/bin/DPS/DpAtMgrBadFunc ConfigFile
/data2/ecs/TS1/CUSTOM/cfg/DpAtBA.CFG *FilesOrDirectories* > *ResultsFile*, press **Return**.
 - The ***FilesOrDirectories*** is a list of source file names or directory names of directories containing source files.
 - The ***ResultsFile*** is the file name for the results that are output.
 - For example, type **/data2/ecs/TS1/CUSTOM/bin/DPS/DpAtMgrBadFunc ConfigFile /data2/ecs/TS1/CUSTOM/cfg/DpAtBA.CFG main.c utils/ > myOutput**, press **Return**. Here, **main.c** is a source file and **utils/** is a directory that contains other source files.
- 4 At the UNIX prompt on the AIT Sun, type **vi *ResultsFile***, press **Return**.

- The **ResultsFile** is the file name for the output results as produced in step 3.
- Any text editor may be used for this procedure step.

Table 10.5.2-1. Checking for Prohibited Functions: Command-Line Version - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	cleartool setview <i>ViewName</i>	press Return
2	cd <i>SrcPathname</i>	press Return
3	\$DPATMGR_HOME/bin/DPS/DpAtMgrBadFunc ConfigFile /ecs/formal/Rel_A/CUSTOM/cfg/DpAtBA_daac.CFG <i>FilesOrDirectories</i> > <i>ResultsFile</i>	press Return
4	vi <i>ResultsFile</i>	press Return

10.6 Checking Process Control Files

Process Control Files (PCFs) should be delivered for each Product Generation Executive (PGE). Only one PCF can be associated with a PGE, however, more than one may be delivered. This procedure describes how to check PCFs for valid syntax and format using the Process Control File Checker, available with a GUI and without.

Section 8.6.1 describes how to use the Process Control File Checker GUI. Section 8.6.2 describes how to use the Process Control File Checker from the command line. Both versions of the checker are functionally identical.

10.6.1 Checking Process Control Files: GUI Version

(Not available on mini-DAAC)

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The SSIT Manager is running.
2. The PCFs to be checked are available, accessible, and have read permissions for the operator.

To check Process Control Files, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **Tools** menu, then choose **Standards Checkers**. Then choose **Process Control File Checker**. See Figure 10.6.1-1 .
 - The Process Control File Checker GUI will be displayed.
 - See Figure 10.6.1-2

- 2 In the **Directories** subwindow, double click on the desired directory.
 - Repeat this step until the directory with the PCF(s) to be checked is displayed in the Files window.
 - Use the **Filter** subwindow to limit which files are displayed.
- 3 Within the **Files** subwindow, click on the PCF to be checked.
 - The file clicked on will be highlighted.
 - Only one PCF can be checked at a time.
 - See Figure 10.6.1-3 .
- 4 Click on the **Check PCF** button.
 - A GUI labeled **PCF Checker Results** will be displayed.
 - Results will be displayed in this window.
- 5 Optionally, click on the **Save** button or on the **Print** button.
 - Choose **Save** to save the results to a file; choose **Print** to have the results printed on the default printer.
 - Choosing **Save** will bring up a GUI labeled **Save To File**. Specify the directory and file name in which to save the results file.
 - Choosing **Print** and then clicking on the **OK** button will send the results to the default printer.
- 6 Click on the **Check Another** button or on the **Quit** button.
 - Choosing **Check Another** allows another PCF to be checked. Repeat steps 2 through 5.
 - Choosing **Quit** causes the Process Control File Checker GUI to disappear and ends the session.

Table 10.6.1-1. Checking Process Control Files: GUI Version - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Tools → Standards Checkers → Process Control File Checker	(No action)
2	directory	click name
3	file(s)	click name(s)
4	Check PCF	click button
5	(Optional) Save Print	click button
6	Check Another Quit	click button

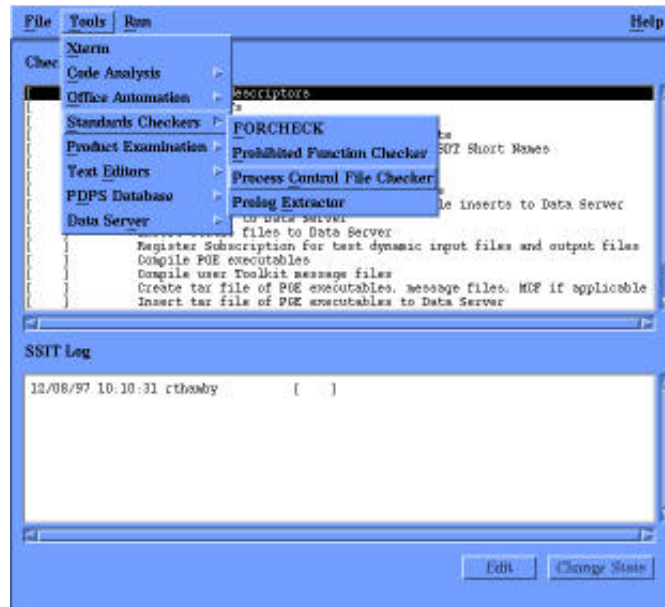


Figure 10.6.1-1. Invoking the Process Control File Checker

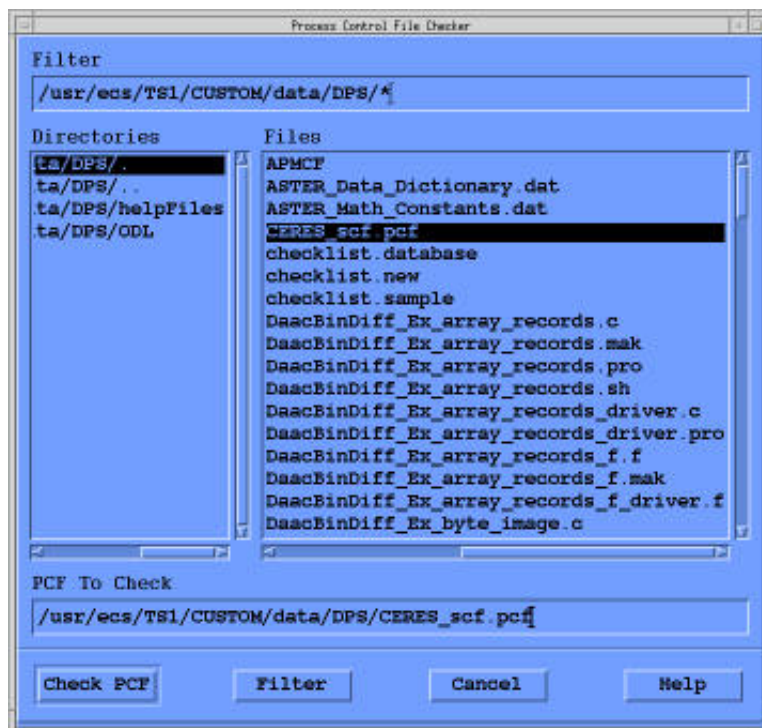


Figure 10.6.1-2. Selecting a File to Check

10.6.2 Checking Process Control Files: Command-Line Version

This procedure describes using the command-line version of the Process Control File Checker to check process control files delivered with the science software.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The PCF files to be checked are available, accessible, and have read permissions for the operator.
2. You will need the command **pccheck.sh**. One way to see if this is available is to type **which pccheck.sh**, press **Return**. If a path is displayed, then the directory is in your path. On the mini-DAAC Sun platform **calahans**, the pathname for the command is `/ecs/formal/TOOLKIT/bin/sun5/pccheck.sh`. In this case, you will have to set a ClearCase view to access that area.

To check Process Control Files, execute the procedure steps that follow:

- 1 If required, at the UNIX prompt on an AIT Sun, type **cleartool setview *ViewName***, press **Return**.
 - The ***ViewName*** is the name of a view allowing the Process Control File(s) to be accessible.
 - This step is only necessary if any of the Process Control Files are in ClearCase (in the VOB under configuration management).
- 2 At the UNIX prompt on AIT Sun, type **cd *PCFpathname***, press **Return**.
 - The ***PCFpathname*** is the full path name to the location of the Process Control File(s) to be checked.
 - The ***PCFpathname*** will be in the ClearCase VOB if the Process Control Files are checked into ClearCase.
- 3 At the UNIX prompt on an AIT Sun, type **/ecs/formal/TOOLKIT/bin/sun5/pccheck.sh -i *PCFfilename* > *ResultsFile***, press **Return**.
 - The ***PCFfilename*** is the full path name (directory and file name) to the Process Control File to check.
 - The ***ResultsFile*** is the file name for the results that are output.
 - The PCF Checker is also available on the SPR SGI machines. The easiest way to access it is to set a SDP Toolkit environment (any will do for purposes here, see Section 9.2) and type **\$PGSBIN/pccheck.sh -i *PCFfilename* > *ResultsFile***, press **Return**.

- 4 At the UNIX prompt on the SPR SGI, type **vi *ResultsFile***, press **Return**.
- The ***ResultsFile*** is the file name for the output results as produced in step 4.
 - Any text editor may be used for this procedure step.

Table 10.6.2-1. Checking Process Control Files: Command-Line Version - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	(Optional) cleartool setview <i>ViewName</i>	press Return
2	cd <i>PCFpathname</i>	press Return
3	pccheck.sh -i <i>PCFfilename</i> > <i>ResultsFile</i>	press Return
4	vi <i>ResultsFile</i>	press Return

10.7 Extracting Prologs

The Project standards and guidelines are contained in the document *Data Production Software and Science Computing Facility (SCF) Standards and Guidelines, Revision A, October 1996* (423-16-01). This ESDIS document mandates that science software delivered to the DAACs to be integrated into the ECS contain prologs in the source files. Prologs are internal documentation containing information about the software. The details are specified in the ESDIS document. Prologs must be at the top of every function, subroutine, procedure, or program module.

This procedure describes using the Prolog Extractor to extract prologs into a file. Note that the prolog extractor only extract the prologs it finds. It does not check the contents of prologs.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The SSIT Manager is running.
2. The source files from which prologs are to be extracted are available, accessible, and have read permissions for the operator.
3. Prologs are delimited by particular delimiters depending on the language type. Delimiters are listed in the Table 8.7-2.
4. Source files have file name extensions shown in Table 8.7-3.

Table 10.7-1. Extracting Prologs - Prolog Delimiters

Language	Type	Delimiter
FORTRAN 77	source	!F77
Fortran 90	source	!F90
C	source	!C
Ada	source	!Ada
FORTRAN 77	include	!F77-INC
Fortran 90	include	!F90-INC
C	include	!C-INC
Any Language	any	!PROLOG
All Languages	The end delimiter is always !END	

Table 10.7-2. Extracting Prologs - File Name Extensions

File Type	File Name Extensions
FORTRAN 77	f, f77, ftn, for, F, F77, FTN, FOR
Fortran 90	f90, F90, f, F
FORTRAN 77/Fortran 90 include	inc, INC
C	c
C/C++ header	h
Ada	a, ada

To extract prologs from delivered source files, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **S**tandards Checkers. Then choose **P**rolog **E**xtractor .

(Note: currently Prolog Extractor is not available on mini-DAAC)

- An xterm (on the AIT Sun) will be displayed within which the Prolog Extractor will be run.
- The Prolog Extractor can also be started from the UNIX prompt. To do this, at the UNIX prompt on the AIT Sun, type
/data2/ecs/TS1/CUSTOM/bin/DPS/DpAtMgrPrologs, press **Return**.

- 2 At the program prompt **Config filename (default defaultConfigFile)?**, press **Return**.

- The default configuration file for this tool will be used.
- The **defaultConfigFile** will be replaced by the full path name and file name of the default configuration file. The file name will be **DpAtPL_daac.CFG** where **daac** will be replaced by one of {GSFC, EDC, LARC, NSIDC}.

- 3 At the **Files(s)? (-h help)** prompt, type in file names and/or directory names containing the files.

- Separate items with spaces.

- If item is a directory (and it exists within the current directory), the contents of the directory will be search recursively for files with valid file name extensions (see Table 10.7-3).
 - Use ./ to indicate the current directory. The extractor will search for all files in the current directory and recursively below.
 - The current directory is the directory from which the SSIT Manager was started.
 - The time needed for the Prolog Extractor could be very long for large numbers of files and directories.
 - When extraction is complete, the message **Output written to file: ./prologs.txt** will be displayed.
- 4 At the program prompt **Hit return for another, 'q <return>' to quit:**, press **Return** to repeat process with another set of source files or type **q** and press **Return** to quit.
- The xterm will disappear.
- 5 At a UNIX prompt on the AIT Sun, type **vi prologs.txt**, press **Return**.
- The extracted prologs file, named **prologs.txt**, will be brought into the editor.
 - Any text editor may be used such as *emacs*. For example, **emacs prologs.txt**, press **Return**.
 - The default location of the **prologs.txt** file is the directory from which the SSIT Manger was invoked.
- 6 Once the extracted prologs file has been examined, exit the editor.
- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.
 - For other editors, refer to that editor's documentation.

Table 10.7-3. Extracting Prologs - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>S</u> tandards Checkers → Prolog <u>E</u> xtractor	(No action)
2	(No entry)	press Return
3	list file names and/or directory names	press Return
4	q Return	press Return
5	vi prologs.txt	press Return
6	:wq	press Return

11. Compiling and Linking Science Software

Science software is developed at independent Science Computing Facilities (SCFs) using the SDP Toolkit. The SDP Toolkit allows science software to be developed for the ECS at independent SCFs. Science software delivered to the DAACs for SSI&T is in the form of source files. In order to be run and tested within the ECS, this science software has to be compiled and linked to one of the SDP Toolkit versions resident at the DAAC to form the binary executables that run within the Product Generation Executives (PGEs).

The procedures which follow describe how to build (compile and link) the science software.

11.1 Updating the Process Control Files (PCFs)

Process Control Files provide the interface between the science software and the production system in the ECS. They contain process information which is needed for each run of the PGE. The process control files delivered to the DAACs for SSI&T were created and used at the SCFs. As such, path names specified in the PCF (and perhaps, file names) will need to be updated since they will not reflect path names in the DAAC environment. The same may be true of file names if they get changed after delivery to the DAAC. Note that these changes are only necessary when the PGE is to be run in an SCF simulated environment (using the SCF version of the SDP Toolkit and run on the command line). When run within the PDPS, the PCF used is generated automatically by the system.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. A PCF for the PGE has been delivered and is available, accessible, and has read permissions.

To update the PCF, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **X**term. Then telnet to the SGI.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SGI.
- 2 If required, at the UNIX prompt on the SGI, type **cleartool setview ViewName**, press **Return**.
 - The **ViewName** is the name of a view allowing the PCF to be accessible.
 - This step is only necessary if the PCF is in ClearCase (in the VOB under configuration management).
- 3 At the UNIX prompt on the Sun or on the SGI, type **cd PCFpathname**, press **Return**.

- The ***PCFpathname*** is the full path name to the location of the PCF. This location will be in the ClearCase VOB if the PCF is under configuration management.
- 4 At the UNIX prompt on the Sun or on the SGI, type **cleartool checkout -nc *PCFfilename***, press **Return**.
- The ***PCFfilename*** is the file name of the PCF that is to be checked out (and later modified). The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check out step.
- 5 Run the Process Control File Checker on the delivered PCF (see Section 10.6).
- This will verify that the delivered PCF is correct before editing.
- 6 At a UNIX prompt on the Sun, type **vi *PCFfilename***, press **Return**.
- The ***PCFfilename*** is the file name of the PCF to update.
 - Any text editor may be used such as *emacs*. For example, **emacs AST02.pcf**, press **Return**.
- 7 In the file, make changes to the default directories specified in each section of the PCF. All path names specified in the PCF must exist on the SGI.
- Each section begins with a line consisting of a ? in the first column followed by a label:
 - ? PRODUCT INPUT FILES
 - ? PRODUCT OUTPUT FILES
 - ? SUPPORT INPUT FILES
 - ? SUPPORT OUTPUT FILES
 - ? INTERMEDIATE INPUT
 - ? INTERMEDIATE OUTPUT
 - ? TEMPORARY I/O
 - Each of the above section heading lines will then be followed (not necessarily immediately; there may be comment lines) by a line that begins with a ! in the first column. These lines specify the default path names for each section.
 - If the line reads:
 - ! ~/runtime
 leave it unchanged. The tilde (~) is a symbol that represents \$PGSHOME.
 - If another path name is listed instead, it will probably need to be changed to a path name that exists at the DAAC on the SGI. When specifying a path name, use an absolute path name, not a relative path name.
- 8 In the file, look for science software specific entries in each section and make changes to the path names (field 3) as necessary. All path names specified in the PCF must exist on the SGI.
- The science software specific entries will have logical IDs (first field) *outside* of the range 10,000 to 10,999.
 - Where necessary, replace the path names in the third field of each entry with the path names appropriate to the DAAC environment.
 - Do not alter file entries that are used by the SDP Toolkit itself. These have logical IDs *in* the range 10,000 to 10,999.
 - For example, if the following entry was found in the PCF:


```
100|L1A.granule|/MODIS/run/input|||1
```


change /MODIS/run/input to the appropriate path name in the DAAC where the file L1A.granule is stored.

- When specifying a path name, use an absolute path name, not a relative path name.
 - Do not include the file name with the path name. The file name belongs in field 2 by itself.
- 9 In the file, verify that the SUPPORT OUTPUT FILES section contains an entry to the shared memory pointer file.
- Look for the entry:

```
10111 | ShmMem | ~/runtime | | | 1
```

The third field may be blank; this will work too.
 - If this entry is not within this section, add it.
- 10 Once changes have been made to the PCF, save the changes and exit the editor.
- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.
 - For other editors, refer to that editor's documentation.
- 11 Again, run the Process Control File Checker on the PCF (see Section 10.6) to make sure that no errors were introduced during editing.
- 12 If the PCF had been checked out of ClearCase, at the UNIX prompt on the SGI, type **cleartool checkin -nc PCFfilename**, press **Return**.
- The **PCFfilename** is the file name of the modified PCF. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check in step.

Table 11.1-1. Updating the Process Control Files (PCFs) - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>X</u> term	telnet to SGI
2	(If necessary) cleartool setview <i>ViewName</i>	press Return
3	cd <i>PCFpathname</i>	press Return
4	cleartool checkout -nc <i>PCFfilename</i>	press Return
5	(No entry)	run Process Control File Checker
6	vi <i>PCFfilename</i>	press Return
7	(No entry)	modify (if necessary) default directory path names in each PCF section
8	(No entry)	modify (if necessary) path names of all science software specific entries
9	(No entry)	verify presence of shared memory pointer file
10	:wq	press Return
11	(No entry)	re-run Process Control File Checker
12	(If necessary) cleartool checkin -nc <i>PCFfilename</i>	press Return

11.2 Setting Up the SDP Toolkit Environment

Science software developed and tested at the SCFs are built using the SCF version of the SDP Toolkit. This version of the SDP Toolkit enables the SCFs to simulate to some extent a DAAC environment. The DAAC version of the SDP Toolkit uses an identical Applications Programming Interface (API) as the SCF version, however, it contains additional hooks into the ECS and provide full ECS functionality. Science software developed at the SCFs with the SCF version of the SDP Toolkit should be able to build and run with the DAAC version of the SDP Toolkit with no changes.

Each SDP Toolkit version (SCF and DAAC) has different flavors depending upon the object type mode and upon the language. The SGI Power Challenge (the machines) run with the IRIX 6.2 operating system. This operating system allows compilers to build binaries in old 32-bit mode, new 32-bit mode, or in 64-bit mode (default). Table 11.2-1 lists the modes available and the corresponding C compiler flags to enable them.

Table 11.2-1. Object Types on the SGI

Object Type	C Compiler Flag Used
Old 32-bit Mode	-32
New 32-bit Mode	-n32
64-bit Mode	-64

In addition, the SDP Toolkit uses different libraries depending upon whether FORTRAN 77 or Fortran 90 source code is being linked. If only C source code is being linked, then either language library can be used.

Given that there are three object types (see Table 11.2-1) and each has two language types (see above paragraph), there are six versions of the DAAC Toolkit at the DAACs. Since the SCF version of the Toolkit is also available, a total of 12 Toolkit versions is available. Table 11.2-2 lists these versions. This table lists the Toolkit version (SCF or DAAC), language type, library object mode, and the home directory of the appropriate SDP Toolkit (represented by the environment variable PGSHOME).

Table 11.2-2. SDP Toolkit Versions at the DAAC

SDP Version	Language Type	Library Object Type	\$PGSHOME	\$PGSBIN
SCF	FORTRAN 77 or C	Old 32-bit mode	\$CUSTOM_HOME/bin/ scf_toolkit_f77/TOOLKIT/	\$CUSTOM_HOME/bin/ scf_toolkit_f77/TOOLKIT/ bin/sgi/
SCF	Fortran 90 or C	Old 32-bit mode	\$CUSTOM_HOME/bin/ scf_toolkit_f90/TOOLKIT/	\$CUSTOM_HOME/bin/ scf_toolkit_f90/TOOLKIT/ bin/sgi/
SCF	FORTRAN 77 or C	New 32-bit mode	\$CUSTOM_HOME/bin/ scf_toolkit_f77/TOOLKIT/	\$CUSTOM_HOME/bin/ scf_toolkit_f77/TOOLKIT/ bin/sgi32/
SCF	Fortran 90 or C	New 32-bit mode	\$CUSTOM_HOME/bin/ scf_toolkit_f90/TOOLKIT/	\$CUSTOM_HOME/bin/ scf_toolkit_f90/TOOLKIT/ bin/sgi32/
SCF	FORTRAN 77 or C	64-bit mode	\$CUSTOM_HOME/bin/ scf_toolkit_f77/TOOLKIT/	\$CUSTOM_HOME/bin/ scf_toolkit_f77/TOOLKIT/ bin/sgi64/
SCF	Fortran 90 or C	64-bit mode	\$CUSTOM_HOME/bin/ scf_toolkit_f90/TOOLKIT/	\$CUSTOM_HOME/bin/ scf_toolkit_f90/TOOLKIT/ bin/sgi6432/
DAAC	FORTRAN 77 or C	Old 32-bit mode	\$CUSTOM_HOME/bin/ daac_toolkit_f77/TOOLKIT /	\$CUSTOM_HOME/bin/ daac_toolkit_f77/TOOLKIT/ bin/sgi/
DAAC	Fortran 90 or C	Old 32-bit mode	\$CUSTOM_HOME/bin/ daac_toolkit_f90/TOOLKIT /	\$CUSTOM_HOME/bin/ daac_toolkit_f90/TOOLKIT/ bin/sgi/
DAAC	FORTRAN 77 or C	New 32-bit mode	\$CUSTOM_HOME/bin/ daac_toolkit_f77/TOOLKIT /	\$CUSTOM_HOME/bin/ daac_toolkit_f77/TOOLKIT/ bin/sgi32/
DAAC	Fortran 90 or C	New 32-bit mode	\$CUSTOM_HOME/bin/ daac_toolkit_f90/TOOLKIT /	\$CUSTOM_HOME/bin/ daac_toolkit_f90/TOOLKIT/ bin/sgi32/
DAAC	FORTRAN 77 or C	64-bit mode	\$CUSTOM_HOME/bin/ daac_toolkit_f77/TOOLKIT /	\$CUSTOM_HOME/bin/ daac_toolkit_f77/TOOLKIT/ bin/sgi64/
DAAC	Fortran 90 or C	64-bit mode	\$CUSTOM_HOME/bin/ daac_toolkit_f90/TOOLKIT /	\$CUSTOM_HOME/bin/ daac_toolkit_f90/TOOLKIT/ bin/sgi64/

\$CUSTOM_HOME is an environment variable set to /usr/ecs/TS1/CUSTOM

The choice of which version of the SDP Toolkit to use depends upon two factors: the test being performed and the version is required by the science software. For running a PGE in a simulated SCF environment (*i.e.* as if at the SCF), a SCF version of the Toolkit should be used (see Section 11). For running a PGE in the fully functional DAAC environment, the DAAC version should be used.

Among the DAAC versions, there are still six choices. Most science software will likely require one of the 32-bit versions. If the science software is written in C only, then the language type does not matter and either language version of the Toolkit may be used. If, however, FORTRAN 77 code is being used (with or without C), then the FORTRAN 77 language version of the DAAC Toolkit must be used. Conversely, if Fortran 90 code is being used (again, with or without C), the Fortran 90 language version of the DAAC Toolkit must be used.

If both FORTRAN 77 and Fortran 90 are being used, the procedure becomes more complex. Under such circumstances, refer to the *Release A SCF Toolkit User's Guide* for details.

This procedure describes how to set up the appropriate SDP Toolkit environment. It involves two basic steps. First, set the SDP Toolkit home directory in the environment variable PGSHOME. The second step is to source (run) the set up script in the appropriate bin directory. This step results in a number of other environment variables getting set that will be needed.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The C shell (or a derivative) is the current command shell.

To check set up a SDP Toolkit environment, execute the procedure steps that follow:

- 1 At the UNIX prompt on the SGI, type **setenv PGSHOME ToolkitPathname**, press **Return**.
 - The **ToolkitPathname** is the home directory of the particular SDP Toolkit version being used. Refer to Table 11.2-2. Note that the setting of PGSHOME shown in this table may differ in your local DAAC.
 - Korn shell users, type **PGSHOME=ToolkitPathname; export PGSHOME**, press **Return**.
- 2 At the UNIX prompt on the SGI, type **source \$PGSHOME/bin/sgiX/pgs-dev-env.csh**, press **Return**.
 - The **sgiX** is one of: **sgi** for 32-bit version of the Toolkit or **sgi64** for 64-bit version of the Toolkit. Refer to the last column of Table 11.2-2 for path names to the file to source.
 - Korn shell users, type **.\$PGSHOME/bin/sgiX/pgs-dev-env.ksh**, press **Return** (note the “dot” and then space at the beginning of this command).

- 3 This step is optional. Edit the file \$HOME/.cshrc and add the line **alias *aliasname* 'setenv PGSHOME *ToolkitPathname*; source \$PGSHOME/bin/*sgiX*/pgs-dev-env.csh; echo "textmessage" '.**
- The ***aliasname*** is the name of the alias. For example, to set up an environment for the DAAC version of the Toolkit for FORTRAN 77 (or C), you might use **DAACf77** as an ***aliasname***.
 - The ***ToolkitPathname*** is the home directory of the particular SDP Toolkit version being used. Refer to Table 11.2-2. Note that the setting of PGSHOME shown in this table may differ in your local DAAC.
 - The ***sgiX*** is one of: **sgi** for 32-bit version of the Toolkit or **sgi64** for 64-bit version of the Toolkit.
 - The ***textmessage*** is a message that will be echoed to the screen signifying that a new Toolkit environment has been set up. It must be enclosed within double quotes ("). An example may be, **"DAAC F77 Toolkit environment is now set."**
 - A complete example (it should be all on one line in the .cshrc file):
**alias DAACf77 'setenv PGSHOME
/\$COSTUM_HOME/bin/daac_toolkit_f77/TOOLKIT; source
\$PGSHOME/bin/sgi/pgs-dev-env.csh; echo "DAAC F77
Toolkit environment is now set" '**
 - Other aliases for other versions of the Toolkit can be set up similarly.

Table 11.2-3. Setting Up the SDP Toolkit Environment - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	setenv PGSHOME <i>ToolkitPathname</i>	press Return
2	source \$PGSHOME/bin/ <i>sgiX</i> /pgs-dev-env.csh	press Return
3	(Optional) alias <i>aliasname</i> 'setenv PGSHOME <i>ToolkitPathname</i> ; source \$PGSHOME/bin/ <i>sgiX</i> /pgs-dev-env.csh; echo "textmessage" '	add line to .cshrc file

11.3 Compiling Status Message Facility (SMF) Files

Status Message Facility (SMF) files are used by the SDP Toolkit to facilitate a status and error message handling mechanism for use in the science software and to provide a means to send log files, informational messages, and output data files to DAAC personnel or to remote users.

Science software making use of the SMF need particular header (include) files when being built and also need particular runtime message files when being run. Both the header and message files are produced by running a SMF "compiler" on a message text file. These message text files should be part of the science software delivery to the DAAC. They typically have a .t file name extension.

This procedure describes how to compile the SMF message text files to produce both the necessary include files and the necessary runtime message files.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The C shell (or a derivative) is the current command shell.

To check compile status message facility (SMF) files, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **X**term. Then telnet to the SGI.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SGI.
 - It is recommended that this procedure begin within a new command shell on the SGI.
- 2 If required, at the UNIX prompt on the SGI, type **cleartool setview *ViewName***, press **Return**.
 - The *ViewName* is the name of a view allowing the SMF files to be accessible.
 - This step is only necessary if any of the SMF files are in ClearCase (in the VOB under configuration management).
- 3 At the UNIX prompt on the SGI, type **setenv PGSHOME *ToolkitPathname***, press **Return**. Then type, source **\$PGSHOME/bin/*sgiX*/pgs-dev-env.csh**, press **Return**.
 - The *ToolkitPathname* is the home directory of the desired SDP Toolkit version (see Section 11.2).
 - The *sgiX* refers to the appropriate processor (see Section 11.2). For example, type **source \$PGSHOME/bin/*sgi*/pgs-dev-env.csh**, press **Return**.
- 4 At the UNIX prompt on the SGI, type **cd *pathname***, press **Return**.
 - The *pathname* is the full path name to the directory containing the SMF text files.
 - The SMF text files will typically have .t file name extensions.
- 5 At the UNIX prompt on the SGI, type **smfcompile -lang -f *textfile.t***, press **Return**.
 - The **-lang** is a flag that indicates for what language to compile. This flag can be one of **-c** to produce C header files, **-f77** to produce FORTRAN 77 include files, and **-ada** to produce Ada include files. The default is for C include files. For example, type **smfcompile -f77 PGS_MODIS_39123.t**, press **Return**.
 - The *textfile* is the file name of the SMF text file delivered with the science software.
 - The SMF text files will typically have .t file name extensions.
 - File names for SMF text files usually have the “seed” value used by the file as part of its file name (*e.g.* PGS_MODIS_39123.t where 39123 is the seed number).
 - Only one such SMF text file can be compiled at a time; wildcards cannot be used.
 - The SMF compiler may be run with the additional flags **-r** and **-i** as in, **smfcompile -f *textfile.t* -r -i**. The **-r** automatically places the runtime message file in the directory given by the environment variable PGSMMSG. The **-i** automatically places the include file in the directory given by the environment variable PGSINC. For example, type **smfcompile -ada -r -i -f PGS_MODIS_39123.t**, press **Return**. Note that the **-f** flag must always be immediately followed by the name of the text file.

- 6 If necessary, at the UNIX prompt on the SGI, type **mv *IncludeFilename* \$PGSINC**, press **Return**. Then, type **mv *RuntimeFilename* \$PGSMSG**, press **Return**.
- This step is only required if either the **-r** or the **-i** flag were not used in step 5.
 - The ***IncludeFilename*** is the name of the include file created in step 5.
 - The ***RuntimeFilename*** is the name of the runtime message file created in step 5.
 - For example, type **mv PGS_MODIS_39123.h \$PGSINC**, press **Return**. And then type, **mv PGS_MODIS_39123 \$PGSMSG**, press **Return**.

Table 11.3-1. Compiling Status Message Facility (SMF) Files - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Tools → Xterm	telnet to SGI
2	(Optional) cleartool setview <i>ViewName</i>	press Return
3	setenv PGSHOME <i>ToolkitPathname</i> source \$PGSHOME/bin/sgiX/pgs-dev-env.csh	press Return press Return
4	cd <i>pathname</i>	press Return
5	smfcompile -lang -f <i>textfile.t</i>	press Return
6	(If necessary) mv <i>IncludeFilename</i> \$PGSINC mv <i>RuntimeFilename</i> \$PGSMSG	press Return press Return

11.4 Building Science Software with the SCF Version of the SDP Toolkit

In order to be tested at the DAAC, science software must be compiled and linked to produce binary executables. These binary executables are then packaged into one or more shell scripts as defined by the science software developer (Instrument Team). These science software packages are the Product Generation Executives (PGEs) delivered to the DAACs during SSI&T. PGEs are the smallest schedulable unit of science software in the ECS.

Building science software into PGEs should be done in accordance with supplied documentation. Such documentation should describe the process in detail. In general, science software deliveries will come with make files or other build scripts to automate the build process.

In general, science software will be built, run, and tested with the SCF version of the SDP Toolkit to ensure that the software has been successfully ported to the DAAC. Once this test has been completed successfully, the science software will be re-built, rerun, and re-tested with the DAAC version of the SDP Toolkit. Only with the DAAC Toolkit can the PGE be run within the ECS.

This procedure describes some general principals that may or may not be applicable to a particular science software delivery for building a PGE with the SCF version of the SDP Toolkit. See Section 11.5 for building a PGE with the DAAC version of the SDP Toolkit.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The C shell (or a derivative) is the current command shell.

To build science software with the SCF version of the SDP Toolkit, be aware of the “typical” procedure steps that follow:

- 1 Read all instructional material supplied with the science software delivery. Such material should be the primary source of information on how to build the science software.
 - Read the *Systems Description* document and the *Operations Manual*. Both of these or their equivalent should be in the delivery.
 - Typically, there will be “readme” files accompanying each PGE in the directory structure, perhaps in a doc directory.
 - Text files (ASCII) may be viewed with the UNIX command, *more* or with the *vi* editor.
 - PostScript documents may be viewed with *ghostview*, which is accessible via the SSIT Manager.
 - PDF formatted documents may be viewed with *acroread*, the Acrobat Reader, also accessible via the SSIT Manager.
 - Documents in Microsoft Word and related formats may be viewed through the Microsoft Windows™ 3.1 emulator. The MS Windows emulator may be accessed from the SSIT Manager.
- 2 From the SSIT Manager, click on the **T**ools menu, then choose **X**term. Then telnet to the SGI.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SGI.
 - It is recommended that this procedure begin within a new command shell on the SGI.
- 3 At the UNIX prompt on the SGI, type **setenv PGSHOME ToolkitPathname**, press **Return**. Then type, source **\$PGSHOME/bin/sgiX/pgs-dev-env.csh**, press **Return**.
 - The **ToolkitPathname** is the home directory of the desired SDP Toolkit version, in this case, an SCF version (see Section 11.2).
 - The **sgiX** refers to the appropriate processor (see Section 11.2). For example, type **source \$PGSHOME/bin/sgi/pgs-dev-env.csh**, press **Return**.
- 4 If make files are in ClearCase, at the UNIX prompt on the SGI, type **cleartool setview ViewName**, press **Return**. Then, **cd pathname**, press **Return**. And **cleartool checkout -nc makefile**, press **Return**.
 - The **ViewName** is the name of a view allowing the make files to be accessible.
 - The **pathname** is the full path name of the directory (in the VOB) where the make file has been checked in.
 - The **makefile** is the name of the make file to examine and possibly modify.
 - This step is only necessary if any of the make files (or build scripts) are in ClearCase (in the VOB under configuration management).

- 5 Examine and alter (if necessary) any make files using any text editor (*vi*, *emacs*).
 - There may be several make files for a particular PGE.
 - Verify that compiler, compiler flag settings, and other environment variable settings are appropriate.
 - The Toolkit set up (from step 3) will set many environment variables which can be used in the make files. To see the current environment variable settings, at the UNIX prompt on the SGI, type **env**, press **Return**.
- 6 Compile any required status message facility (SMF) files and place the header file(s) in the proper directory for building. See Section 11.3.
- 7 Verify that the directory structure for the PGE source files matches the directory structure expected by the make files or build scripts.
 - Deliveries may come with install scripts that place files into various directories according to some predefined structure.
- 8 If necessary, at the UNIX prompt on the SGI, type **cleartool checkout -nc *filename***, press **Return**.
 - The *filename* is the file name of the executable, object file, or make file to be checked out of ClearCase. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check out step.
 - Note that checking in executable or object files is *not* recommended in the first place.
- 9 Build the software in accordance with instructions delivered.
 - Science software deliveries may come with a single, top-level script to do the entire build or the build process could involve a series of steps, each of which should be described fully in the delivered documentation.
- 10 If necessary, at the UNIX prompt on the SGI, type **cleartool checkin *filename* -nc**, press **Return**.
 - The *filename* is the file name of the executable, object file, or make file to be checked into ClearCase. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check in step.
 - Note that checking in executable or object files is *not* recommended.

Table 11.4-1. Building Science Software with the SCF Version of the SDP Toolkit - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	(No entry)	read all documentation
2	<u>T</u> ools → <u>X</u> term	telnet to SGI
3	setenv PGSHOME <i>ToolkitPathname</i> source \$PGSHOME/bin/sgiX/pgs-dev-env.csh	press Return press Return
4	(If necessary) cleartool setview <i>ViewName</i> cd <i>pathname</i> cleartool checkout -nc <i>makefile</i>	press Return press Return press Return
5	(No entry)	examine/alter make files
6	smfcompile -lang -f <i>textfile.t</i>	press Return
7	(No entry)	verify directory structure
8	(If necessary) cleartool checkout -nc <i>filename</i>	press Return
9	(No entry)	build the software according to documentation
10	(If necessary) cleartool checkin <i>filename</i> -nc	press Return

11.5 Building Science Software with the DAAC Version of the SDP Toolkit

In general, science software will be built, run, and tested with the SCF version of the SDP Toolkit to ensure that the software has been successfully ported to the DAAC. Once this test has been completed successfully, the science software will be re-built, rerun, and re-tested with the DAAC version of the SDP Toolkit. Only with the DAAC Toolkit can the PGE be run within the ECS.

This procedure describes some general principals that may or may not be applicable to a particular science software delivery for building a PGE with the DAAC version of the SDP Toolkit. See Section 11.4 for building a PGE with the SCF version of the SDP Toolkit.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The C shell (or a derivative) is the current command shell.

To build science software with the DAAC version of the SDP Toolkit, be aware of the “typical” procedure steps that follow:

- 1 Read all instructional material supplied with the science software delivery. Such material should be the primary source of information on how to build the science software.
 - Read the *Systems Description* document and the *Operations Manual*. Both of these or their equivalent should be in the delivery.
 - Typically, there will be “readme” files accompanying each PGE in the directory structure, perhaps in a doc directory.
 - Text files (ASCII) may be viewed with the UNIX command, *more* or with the *vi* editor.
 - PostScript documents may be viewed with *ghostview*, which is accessible via the SSIT Manager.
 - PDF formatted documents may be viewed with *acroread*, the Acrobat Reader, also accessible via the SSIT Manager.
 - Documents in Microsoft Word and related formats may be viewed through the Microsoft Windows™ 3.1 emulator. The MS Windows emulator may be accessed from the SSIT Manager.
- 2 From the SSIT Manager, click on the **T**ools menu, then choose **X**term. Then telnet to the SGI.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SGI.
 - It is recommended that this procedure begin within a new command shell on the SGI.
- 3 At the UNIX prompt on the SGI, type **setenv PGSHOME ToolkitPathname**, press **Return**. Then type, source **\$PGSHOME/bin/sgiX/pgs-dev-env.csh**, press **Return**.
 - The **ToolkitPathname** is the home directory of the desired SDP Toolkit version, in this case, a DAAC version (see Section 11.2).
 - The **sgiX** refers to the appropriate processor (see Section 11.2). For example, type **source \$PGSHOME/bin/sgi/pgs-dev-env.csh**, press **Return**.
- 4 If make files are in ClearCase, at the UNIX prompt on the SGI, type **cleartool setview ViewName**, press **Return**. Then, **cd pathname**, press **Return**. And **cleartool checkout -nc makefile**, press **Return**.
 - The **ViewName** is the name of a view allowing the make files to be accessible.
 - The **pathname** is the full path name of the directory (in the VOB) where the make file has been checked in.
 - The **makefile** is the name of the make file to examine and possibly modify.
 - This step is only necessary if any of the make files (or build scripts) are in ClearCase (in the VOB under configuration management).
- 5 Examine and alter (if necessary) any make files using any text editor (*vi*, *emacs*). If the software had already been built and tested with the SCF version of the SDP Toolkit, this step may be unnecessary.
 - There may be several make files for a particular PGE.
 - Verify that compiler, compiler flag settings, and other environment variable settings are appropriate.

- The Toolkit set up (from step 3) will set many environment variables which can be used in the make files. To see the current environment variable settings, at the UNIX prompt on the SGI, type **env**, press **Return**.
- 6 Compile any required status message facility (SMF) files and place the header file(s) in the proper directory for building. See Section 11.3.
 - 7 Verify that the directory structure for the PGE source files matches the directory structure expected by the make files or build scripts.
 - Deliveries may come with install scripts that place files into various directories according to some predefined structure.
 - 8 If necessary, at the UNIX prompt on the SGI, type **cleartool checkout -nc *filename***, press **Return**.
 - The *filename* is the file name of the executable, object file, or make file to be checked out of ClearCase. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check out step.
 - Note that checking in executable or object files is *not* recommended in the first place.
 - 9 Build the software in accordance with instructions delivered.
 - Science software deliveries may come with a single, top-level script to do the entire build or the build process could involve a series of steps, each of which should be described fully in the delivered documentation.
 - 10 If necessary, at the UNIX prompt on the SGI, type **cleartool checkin *filename* -nc**, press **Return**.
 - The *filename* is the file name of the executable, object file, or make file to be checked into ClearCase. The **-nc** flag means “no comment”; it suppresses ClearCase from prompting for a comment to be associated with the check in step.
 - Note that checking in executable or object files is *not* recommended.

**Table 11.5-1. Building Science Software with the DAAC Version of the SDP Toolkit
- Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	(No entry)	read all documentation
2	<u>T</u> ools → <u>X</u> term	telnet to SGI
3	setenv PGSHOME <i>ToolkitPathname</i> source \$PGSHOME/bin/sgiX/pgs-dev-env.csh	press Return press Return
4	(If necessary) cleartool setview <i>ViewName</i> cd <i>pathname</i> cleartool checkout -nc <i>makefile</i>	press Return press Return press Return
5	(No entry)	examine/alter make files
6	smfcompile -lang -f <i>textfile.t</i>	press Return
7	(No entry)	verify directory structure
8	(If necessary) cleartool checkout -nc <i>filename</i>	press Return
9	(No entry)	build the software according to documentation
10	(If necessary) cleartool checkin <i>filename</i> -nc	press Return

This page intentionally left blank.

12. Running a PGE in a Simulated SCF Environment

Science software delivered to the DAACs for SSI&T was developed and tested at individual SCFs using the SCF version of the SDP Toolkit. Before linking the software with the DAAC version of the Toolkit and integrating it with the ECS, it is prudent to first link the software to the SCF version of the Toolkit and run it as it was run at the SCF. This type of testing can reveal problems associated with the process of porting the software to another platform whose architecture may be quite different from the one on which the software was developed.

A simulated SCF environment means that the software is built using the SCF version of the Toolkit and is run from the UNIX command line. The Planning and Data Processing System (PDPS) and the Data Server are not involved.

The procedures which follow describe how to run the science software in a simulated SCF environment.

12.1 Setting Up the Environment for Running the PGE

Running a PGE that has been built with the SCF version of the SDP Toolkit requires some environment set up as it does at the SCF. This procedure describes how to set up a simulated SCF environment. Once the environment has been set up, the PGE can be run as described in Section 12.2.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The Process Control File (PCF) exists and has been tailored for the DAAC environment (Section 9.1).
2. The C shell or a derivative (*e.g.* T shell) is the current user shell.

To set up an environment for running the PGE, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **X**term. Then telnet to the SPR SGI.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the SPR SGI.
 - It is recommended that this procedure begin within a new command shell on the SPR SGI.
- 2 At the UNIX prompt on the SPR SGI, type **setenv PGSHOME ToolkitPathname**, press **Return**. Then type, source **\$PGSHOME/bin/sgiX/pgs-dev-env.csh**, press **Return**.

- The ***ToolkitPathname*** is the home directory of the desired SDP Toolkit version, in this case, an SCF version (see Section 9.2).
 - The ***sgiX*** refers to the appropriate processor (see Section 9.2). For example, type **source \$PGSHOME/bin/sgi/pgs-dev-env.csh**, press **Return**.
- 3 At the UNIX prompt on the SPR SGI, type **setenv PGS_PC_INFO_FILE *PCFpathname/PCFfilename***, press **Return**.
- The ***PCFpathname*** is the full path name to the location of the Process Control File (PCF) to be associated with this PGE.
 - The ***PCFfilename*** is the file name of the PCF.
 - For example, **setenv PGS_PC_INFO_FILE /disk2/PGE32/PCF/PGE32.pcf**, press **Return**.
- 4 Optionally, at the UNIX prompt on the SPR SGI, type **rm *LogPathname/LogFilename***, press **Return**.
- The ***LogPathname*** is the full path name to the location of the PGE log files for this PGE.
 - The ***LogFilename*** is the file name of the PGE log file to remove from a previous run of the same PGE. PGE log files can be Status, User, or Report.
 - The ***LogFilename*** may use wildcard characters to remove all of the log files at the same time.
 - This step is optional. If log files from a previous run of the same PGE are not removed, they will be appended with the information from the current run.
 - The environment will then be set up. Continue on to Section 12.2.
- 5 If necessary, set any other shell environment variables needed by the PGE by sourcing the appropriate scripts or setting them on the command line.
- For example, for a PGE requiring IMSL, at the UNIX prompt on the SPR SGI, type **source /usr/ecs/<mode>/COTS/imsl/vni/ipt/bin/iptsetup.cs**, press **Return**.
 - For some PGEs, the environment variables to be set will be specified in the documentation or the files to source will be supplied in the delivery. Refer to documentation included in the delivery.

Table 12.1-1. Setting Up the Environment for Running the PGE - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>X</u> term	telnet to SPR SGI
2	setenv PGS_PC_INFO_FILE <i>PCFpathname/PCFfilename</i>	press Return
3	setenv PGSHOME <i>ToolkitPathname</i> source \$PGSHOME/bin/sgiX/pgs-dev-env.csh	press Return press Return
4	(Optional) rm <i>LogPathname/LogFilename</i>	press Return
5	(If necessary) (No entry)	set any additional environment variables needed

12.2 Running and Profiling the PGE

Profiling a PGE refers to the process of gathering information about the runtime behavior of a PGE. The information includes the wall clock time, user time and system time devoted to the PGE; the amount of memory used; the number of page faults; and the number of input and output blocks.

The Planning and Data Processing System (PDPS) database must be populated with the above information when the PGE is registered with the PDPS during the integration phase of SSI&T. This information may be delivered with the PGE or it may need to be determined at the DAAC during SSI&T. This procedure addresses the latter need.

Note that profiling, as used here, does not involve altering the binary executable to produce instrumented code.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The PGE has been built successfully with the SCF version of the SDP Toolkit (Section 11.4).
2. The required SMF runtime message files have been produced and placed in the correct locations (Section 11.3).
3. The Process Control File (PCF) exists and has been tailored for the DAAC environment (Section 11.1).
4. The required environment for running the PGE has been set up (Section 12.1).
5. The required input files are available and accessible.
6. The C shell or a derivative (*e.g.* T shell) is the current user shell.

To run and profile the PGE, execute the procedure steps that follow:

- 1 At the UNIX prompt on the SPR SGI in the window containing the set up environment from Section 12.1, type **cd *PGEbinPathname***, press **Return**.
 - The *PGEbinPathname* is the full path name of the directory containing the built PGE binary executable. For example, **cd /disk2/PGE32/bin/**, press **Return**.
- 2 At the UNIX prompt on the SPR SGI, type **DpPrRusage PGE >& ResultsOut**, press **Return**.
 - The *PGE* is the name given to the PGE binary executable.
 - The *ResultsOut* is the file name in which to capture the profiling results as well as any messages from standard output (stdout) and standard error (stderr) that may be produced by the running PGE. Note that PGEs should *not* write to stdout or stderr.
 - The **DpPrRusage** is the profiling program that outputs information about the runtime behavior of the PGE.

- Depending upon the PGE, it may take some time before the UNIX prompt returns.
- 3 At the UNIX prompt on the SPR SGI, type **echo \$status**, press **Return**.
- The **\$status** is an environment variable that stores the exit status of the previous program run, in this case, the PGE.
 - A status of zero indicates success; a status of non-zero indicates an error of some kind.
 - The meaning of a non-zero exit status should be documented and included with the DAPs.
 - This command must be run *immediately* after the **DpPrRusage** command.
- 4 At the UNIX prompt on the SPR SGI, type **vi ResultsOut**, press **Return**.
- The **ResultsOut** is the file name under which the profiling output was saved. Other output of the PGE may also be in this file.
 - The **DpPrRusage** results may then be recorded and used when the PGE is registered in the PDPS.
 - Any text editor/viewer may be used.

Table 12.2-1. Running and Profiling the PGE - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<code>cd PGEbinPathname</code>	press Return
2	<code>DpPrRusage PGE >& ResultsOut</code>	press Return
3	<code>echo \$status</code>	press Return
4	<code>vi ResultsOut</code>	press Return

13. PGE Registration and Test Data Preparation

The integration of science software with the ECS requires that information about the Product Generation Executives (PGEs) be made known to the PDPS in its database. In addition, the PGEs themselves and the test files that they use (both input and output) need to be placed on the Data Server. These steps must be accomplished before the science software can be run and tested within the ECS.

13.1 PGE Registration

The procedures in this section describe how to register a new PGE with the ECS. This involves updating the PDPS database with information needed to plan, schedule, and run the PGE. Section 13.1.1 describes how to begin the PGE registration process by creating the necessary ODL files needed for each input and output ESDT used by the PGE. Section 13.1.2 describes how to create an ODL file from the delivered PCF and update the PDPS database with this PGE information. Finally, Section 13.1.3 describes how to supply operational information (resource requirements and runtime statistics) to the PDPS database. This is the last step in the PGE registration process. The order in which these procedures are done is important and should be done as indicated.

13.1.1 Updating the PDPS Database with ESDT Metadata

In order to update the PDPS Database with ESDT metadata, the metadata must first be prepared in ODL files, one for each ESDT associated with a particular PGE. Then the SSIT Science Update program must be run (Section 13.1.2). The Procedures below are to be followed for each input and output file for a given PGE.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The environment variable `DPAT_ESDT_SCIENCE_MD` points to a directory used for containing the PDPS ESDT metadata ODL files. Nominally, this is `/usr/ecs/<mode>/CUSTOM/data/DPS/ODL`

To update the PDPS Database with ESDT Metadata, execute the procedure steps that follow:

- 1 Determine ShortName for the ESDT corresponding to the given file on the Data Server. One place to look for ESDT names is the Earth Sciences Online Directory.
- 2 At a UNIX prompt on an AIT Sun, type `ls $DPAT_ESDT_SCIENCE_MD/ESDT_ShortName.odl`, press **Return**.

- The **\$DPAT_ESDT_SCIENCE_MD** is an environment variable containing the full path name to the location of existing ESDT ODL files.
 - The **ESDT_ShortName.odl** is the file name of the ESDT ODL file you are looking for where *ShortName* is the ESDT's ShortName. If a file for the desired ESDT is listed, then it has already been prepared and this procedure can be exited now.
 - For example, if the desired ESDT has the ShortName MOD03, type **ls \$DPAT_ESDT_SCIENCE_MD/ESDT_MOD03.odl**, press **Return**.
 - If the desired file is *not* listed, continue on to step 3.
- 3 At a UNIX prompt on the AIT Sun, type **cd WorkingPathname**, press **Return**.
- The **WorkingPathname** is the full path name to a working directory for which the user has write permissions.
 - For example, **cd /home/jdoe/working/**, press **Return**.
- 4 At a UNIX prompt on the AIT Sun, type **cp /usr/ecs/<mode>/CUSTOM/data/DPS/ESDT_ODL.template ESDT_ShortName.odl**, press **Return**.
- For <mode> enter the mode you are working in, for example **OPS** or **TS1**.
 - The **ESDT_ShortName.odl** is the file name of the ESDT ODL file to be created.
 - This command copies a template ESDT ODL file to the ESDT ODL file to be created. The template is well commented.
 - For example, type **cp /usr/ecs/TS1/CUSTOM/data/DPS/ESDT_SHRTNAME.odl.tpl ESDT_ca1182.odl**, press **Return**.
 - The **ESDT_ShortName.odl** file naming convention *must* be observed.
- 5 At a UNIX prompt on the AIT Sun, type **vi ESDT_ShortName.odl**, press **Return**.
- The **ESDT_ShortName.odl** represents the file name of the ESDT ODL template file created in step 4.
 - Any text editor may be used such as *emacs*. For example, **emacs ESDT_MOD03.odl**, press **Return**.
- 6 In the file, add required metadata to the ODL template.
- Use the internal documentation contained in the ODL file (from the original template) to aid in populating with metadata.
 - Note that the ShortName specified within the file must match the ShortName of the file name itself.
 - In addition, the ShortNames used in the PDPS PGE metadata ODL file (see Section 13.1.2) must match the ShortNames in these files.
- 7 Save the changes made to the ESDT metadata ODL file and exit the editor.
- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.
 - For other editors, refer to that editor's documentation.
- 8 Repeat steps 1 through 7 for each ESDT required by a particular PGE. When all ESDT metadata ODL files have been completed, continue on to Section 13.1.2.

Table 13.1.1-1. Updating the PDPS Database with ESDT Metadata - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	determine ESDT ShortName on the Data Server	(No action)
2	ls \$DPAT_ESDT_SCIENCE_MD/ESDT_ <i>ShortName</i> .odl	press Return
3	cd <i>WorkingPathname</i>	press Return
4	cp /usr/ecs/<mode>/CUSTOM/data/DPS/ ESDT_ODL.template ESDT_ <i>ShortName</i> .odl	press Return
5	vi ESDT_ <i>ShortName</i> .odl	press Return
6	add required metadata to file	(No action)
7	:wq	press Return
8	(No entry)	repeat steps 1-7 for all ESDTs

13.1.2 Updating the PDPS Database with PGE Metadata

In order to update the PDPS Database with PGE metadata, the ESDT metadata ODL files must first be prepared for each ESDT required by the PGE (Section 13.1.1). This section describes how to perform the next step, running the SSIT Science Update program.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The SSIT Manager is running.
2. The environment variable DPAT_PGE_SCIENCE_MD points to a directory used for containing the PDPS PGE metadata ODL files. Nominally, this is /usr/ecs/<mode>/CUSTOM/data/DPS/ODL.
3. The required PDPS ESDT metadata ODL files have been prepared and placed in the directory pointed to by DPAT_ESDT_SCIENCE_MD (see Section 13.1.1).

To update the PDPS Database with PGE Metadata, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **PDPS Database** and then **PCF ODL Template**.
 - An xterm in which DpAtCreateOdlTemplate.sh is running will be displayed.
 - Alternatively, the same tool can be invoked by typing at a UNIX prompt on an AIT Sun **DpAtCreateOdlTemplate.sh**, press **Return**.
- 2 At the program prompt **Configuration Filename (default defaultConfigFile)?**
 - Type in **../.. /cfg/ defaultConfigFile** and press **Return**.

- The *defaultConfigFile* will be replaced by the full path name and file name of the default configuration file. The file name will be DpAtCS_*daac*.CFG where *daac* will be replaced by one of {GSFC, EDC, LARC, NSIDC}.
- 3 At the program prompt **ECS mode of operation (enter for default: defaultMode)?**, type *mode*, press **Return** or just press **Return** if the default shown is correct.
 - The *mode* refers to the database used and will typically be **OPS** or **TS1**.
 - 4 At the program prompt **Process Control file name?**, type *PCFpathname/PCFfilename*, press **Return**.
 - The *PCFpathname* is the full path name to the location of the PCF. If not specified, the directory from which the SSIT Manager was run will be assumed.
 - The *PCFfilename* is the file name of the PCF.
 - 5 At the program prompt **PGE name (max 12 characters)?**, type *PGEname*, press **Return**.
 - The *PGEname* is the name of the PGE that will be registered.
 - 6 At the program prompt **PGE version (max 5 characters)?**, type *PGEversion*, press **Return** or just press **Return** if the default shown is correct.
 - The *PGEversion* is the version of the PGE that will be registered.
 - 7 At the prompt **PGE Profile ID (0 for Null, max 99)** use the default value of 1.
 - After a brief time, the message “Successfully created ODL template file” should be displayed if the task was successful.
 - The program will output a file with the filename **PGE_*PGEname*#*PGEversion*#*ProfileID*.tpl**.
 - For example, if the PGE name was **MOD35**, and the version and profile ID were both **1** this output file will be named **PGE_MOD35#1#01.tpl**.
 - 8 At the program prompt **Hit return to run again, 'q <return>' to quit:**, press **Return** to repeat process with another PCF or type **q** and press **Return** to quit.
 - The xterm will disappear.
 - 9 At a UNIX prompt on an AIT Sun, type **cd *SSITrunPathname***, press **Return**.
 - The *SSITrunPathname* is the full path to the directory from which the SSIT Manager was run. This will be the directory where the file **PGE_*PGEname*#*PGEversion*#*ProfileID*.tpl** will reside.
 - 10 At a UNIX prompt on the AIT Sun, type **cp *PGE_*PGEname*#*PGEversion*#*ProfileID*.tpl* *PGE_*PGEname*#*PGEversion*#*ProfileID*.odl***, press **Return**.
 - The *PGE_*PGEname*#*PGEversion*#*ProfileID*.tpl* is the file name of the ODL template file created in step 6.
 - The *PGE_*PGEname*#*PGEversion*#*ProfileID*.odl* is the file name of a copy which can be safely edited. This file name convention must be used.
 - 11 At a UNIX prompt on the AIT Sun, type **vi *PGE_*PGEname*#*PGEversion*#*ProfileID*.odl***, press **Return**.
 - The *PGE_*PGEname*#*PGEversion*#*ProfileID*.odl* is the file name of the copy created in step 10.

- Any text editor may be used such as *emacs*. For example, **emacs MOD35#1#01.odl**, press **Return**.
- 12 In the file, add required metadata to the ODL template.
- For an explanation of what metadata is required, see file `/usr/ecs/<mode>/CUSTOM/data/DPS/PGE_ssit#1.tpl`.
 - Note that the ShortNames typed into this file must each have a corresponding PDPS ESDT metadata ODL file (sec. 13.1.1).
 - Note also that two PCF entry objects may need to be deleted corresponding to logical ID 10999. Unless the default constants conversion file and the single MCF file, respectively, are to be overridden then these entries in the ODL file should be deleted. Delete the entire objects from the file. Further, adjust the `CLASS=n` number so that the remaining PCF entry objects have class numbers in numerical order.
 - For PCF entries corresponding to MCFs, add to the object “`SCIENCE_GROUP=Mn`” where *n* is an integer (1 for the first MCF, 2 for the second, etc.).
 - All objects corresponding to output ESDTs must have the `SCIENCE_GROUP` and `YIELD` set.
 - All objects corresponding to static input ESDTs must have the `SCIENCE_GROUP` set, but *not* the `YIELD` (leave out). Objects corresponding to *dynamic* input ESDTs should not have the `SCIENCE_GROUP` set.
 - See Appendix E for example PCF and corresponding PGE ODL file.
- 13 Save the changes made to the ODL template file and exit the editor.
- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.
 - For other editors, refer to that editor’s documentation.
- 14 From the SSIT Manager, click on the **Tools** menu, then choose **PDPS Database** and then **SSIT Science Metadata Update**.
- An xterm in which `DpAtPdpsDbUpdateScience.sh` is running will be displayed.
 - Alternatively, the same tool can be invoked by typing at a UNIX prompt on an AIT Sun **DpAtPdpsDbUpdateScience.sh**, press **Return**.
- 15 At the program prompt **Configuration Filename (default *defaultConfigFile*)?**
- Type in `../..cfg/ defaultConfigFile` and press **Return**.
 - The *defaultConfigFile* will be replaced by the full path name and file name of the default configuration file. The file name will be `DpAtDS_daac.CFG` where *daac* will be replaced by one of {GSFC, EDC, LARC, NSIDC}.
- 16 At the program prompt **ECS mode of operation (default *defaultMode*)?**, type *mode*, press **Return** or just press **Return** if the default shown is correct.
- The *mode* refers to the database used and will typically be **OPS** or **TS1**.
- 17 At the program prompt **PGE name (max 12 characters)?**, type *PGEname*, press **Return**.
- The *PGEname* is the name of the PGE that will be registered. This name must match the PGE name specified in step 5.

- 18 At the program prompt **PGE version (default 1)?**, type *PGEversion*, press **Return** or just press **Return** if the default shown is correct.
 - The *PGEversion* is the version of the PGE that will be registered. This version must match the PGE version specified in step 6.
- 19 At the program prompt **PGE Profile ID (0-99, 0 means null)? (enter for default: 1)**, press **Return** for default or enter any valid ID.
 - The PDPS database will then be updated with the information contained in the file **PGE_PGEname#PGEversion#ProfileID.odl**
- 20 At the program prompt **Hit return to run again, q <return> to quit:**, press **Return** to update the PDPS database with another PGE ODL metadata file or type **q** and press **Return** to quit.
 - If you make a mistake entering any values, press **Return** here; your previous enteries are restored as defaults and you won't have to retype them.

Table 13.1.2-1. Updating the PDPS Database with PGE Metadata - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>P</u> DPS Database → <u>P</u> CF ODL Template	(No action)
2	<i>Configuration Filename</i>	press Return
3	<i>ECS mode of operation</i>	press Return
4	<i>PCFpathname/PCFfilename</i>	press Return
5	<i>PGEname</i>	press Return
6	<i>PGEversion</i>	press Return
7	<i>PGE Profile ID</i>	press Return
8	q Return	press Return
9	cd <i>SSITrunPathname</i>	press Return
10	cp PGE_PGEname#PGEversion#ProfileID.tpl PGE_PGEname#PGEversion#ProfileID.odl	press Return
11	vi PGE_PGEname#PGEversion#ProfileID.odl	press Return
12	add required metadata to file	(No action)
13	:wq	press Return
14	<u>T</u> ools → <u>P</u> DPS Database → <u>S</u> SIT Science Metadata Update	(No action)
15	<i>Configuration Filename</i>	press Return
16	<i>mode</i>	press Return
17	<i>PGEname</i>	press Return
18	<i>PGEversion</i>	press Return
19	<i>ProfileID</i>	press Return
20	q Return	press Return

13.1.3 Updating the PDPS Database with Operational Metadata

The SSIT version of the PDPS database is initialized and updated with SSIT Operational Metadata so that the Planning and Processing Subsystem can schedule and run PGEs. Here, PDPS Operational Metadata refers to PGE information which is supplied to the DAAC/SSIT Operator and may change frequently.

The operator enters this data directly into the SSIT Operational Metadata Update GUI. The program then writes the data directly to the SSIT version of the PDPS database.

Before running the SSIT Operational Metadata Update from either the SSIT Manager or from the command line, you must first update the PDPS with PGE metadata (see Section 13.1.2) and with ESDT metadata (see Section 13.1.1). In addition, to get initial PGE Performance data which will be entered into the GUI, you need to run the profiling utility, EcDpPrRusage on the PGE or have the information on profiling provided (see Section 12.2).

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The required UNIX environment variables have been set.
2. The PGE and ESDT metadata have been updated to the PDPS database for this PGE (Sections 13.1.1 and 13.1.2).
3. The SSIT Manager's internal PCF entry that calls this tool has the correct database name in the argument list. That is, the relevant entry looks like:

```
630 | DpCAAtMgrSSITOpnlMdUp | EcDpAtOpDbGui ConfigFile  
/usr/ecs/<mode>/CUSTOM/cfg/EcDpAtDO_daac.CFG ecs_mode  
mode
```

where *daac* is one of {GSFC, EDC, LARC, NSIDC} and the *mode* refers to the correct database (typically set to TS1). For example,

```
630 | DpCAAtMgrSSITOpnlMdUp | DpAtOpDbGui ConfigFile  
/usr/ecs/TS1/CUSTOM/cfg/DpAtDO_GSFC.CFG ecs_mode TS1
```

To update the SSIT version of the PDPS database with operational metadata, execute the steps that follow:

- 1 From the SSIT Manager, click on the **Tools** menu, then choose **PDPS Database** and then **SSIT Opnl Metadata Update**.
 - The PDPS/SSIT Database Update GUI will be displayed.
- 2 Click on the radio button labeled **NEW PGE** in the lower left quadrant.
 - The PGE that you are working on should appear in the subwindow labeled **PGE Names** along with its version number in the subwindow labeled **PGE Versions**.

- 3 In the subwindow labeled **PGE Names**, click on a PGE name. Then in the subwindow labeled **PGE Versions**, click on the PGE version for that PGE. Then click on the button labeled **EDIT**.
 - The PGE name and version will be highlighted when you click on them.
 - The page tabs **PROFILE**, **RUNTIME**, and **ESDT** will change from grey (indicating disabled) to black (indicating enabled).
 - To see the contents of PGE Metadata, click on the button labeled **DISPLAY** and then click on the button labeled **DONE**.
 - If the PGE name and/or version does not appear in the lists, it means that the updating of the PDPS database with PGE metadata was not successful.
- 4 Click on the **PROFILE** page tab.
 - The Profile page will be displayed.
- 5 In the fields under the label **Performance Statistics**, enter the information specified.
 - In the field labeled **Wall clock time**, enter the amount of wall clock time it takes for one execution of the PGE, in seconds. The tab **PROFILE** will change from black (indicating enabled) to red (indicating database needs to be updated by APPLY button).
 - In the field labeled **CPU time (user)**, enter the so-called *user* time of the PGE, in seconds. This value should come from profiling the PGE (see Section 12.2).
 - In the field labeled **Max memory used**, enter the maximum amount of memory used by the PGE, in megabytes (MB). This value should come from profiling the PGE (see Section 10.2).
 - In the fields labeled **Block input ops** and **Block output ops**, enter the integer number of block inputs and block outputs, respectively. These values should come from profiling the PGE (see Section 12.2).
 - In the field labeled **Swaps**, enter the integer number of page swaps from the PGE. This value should come from profiling the PGE (see Section 12.2).
 - In the field labeled **Page faults**, enter the integer number of page faults from the PGE. This value should come from profiling the PGE (see Section 12.2).
- 6 In the fields under the label **Resource Requirements**, enter the information specified.
 - In the field labeled **Max. DISK SPACE used during PGE run**, enter the maximum amount of disk used by the PGE during execution, in megabytes (MB). Space should be allowed for the executable(s), input files, output files, ancillary files, static files, MCFs, and the PCF. (This number should also be in the PGE metadata ODL file; yes, there is duplication here.)
 - Click on the radio button labeled **Proc. String** (if not already clicked on).
 - A list of processing strings should appear in the scrollable window to the left of the two radio buttons **Proc. String** and **Computer Name**. Nominally, only one item should be listed and should be highlighted.
 - In the field labeled **Number of CPUs**, the number 1 should appear.
- 7 Once the fields on the **PROFILE** page have been completed, click on the **APPLY** button.
 - This will update the PDPS database with the information just entered. The tab **PROFILE** will change from red (indicating database needs to be updated) to black (indicating enabled).

- An information box will be displayed; click on **Ok**.
 - To start over, click on the **RESET** button. This will clear all fields.
- 8 Click on the **File** menu and select **Exit**.
- This will end the session with PDPS/SSIT Database Update and the GUI will disappear.

Table 13.1.3-1. Updating the PDPS Database with Operational Metadata - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Tools → PDPS Database → SSIT Opnl Metadata Update	(No action)
2	NEW PGE	click button
3	PGE name PGE version EDIT	click click click button
4	PROFILE	click tab
5	Wall clock time CPU time (user) Max memory used Block input ops Block output ops Swaps Page faults	(No action)
6	Max. DISK SPACE used during PGE run Proc. String	
7	APPLY	click button
8	File → Exit	(No action)

13.2 Test Data Preparation

This section describes how to prepare test data for use by registered PGEs. When PGEs are first delivered to the DAAC and registered within the PDPS, they will typically be run in isolation. That is, they will be run without any PGE dependencies. For this testing to be possible, test input data granules required by the PGE need to be pre-Inserted to the Data Server.

Data granules can be *dynamic* or *static*. Dynamic data granules are those whose temporal locality differs for each instance of the granule. Examples of dynamic granules are Level 0, Level 1, and Level 2 data sets. Static data granules are those whose temporal locality is static over long periods of time. Examples of static granules are calibration files which may only change with a new version of a PGE. For any granule to be Inserted to the Data Server, a Target MCF is needed (also known as an ASCII metadata ODL file or a .met file).

In the actual production environment, a Target MCF is produced by the PGE during execution. Thus, the data granule can be Inserted. In isolation testing of a PGE, however, the inputs needed

by it will not have been Inserted by a previous PGE in the chain. This Insertion must be done manually. Section 13.2.2 describes how to use the Source MCF for a dynamic data granule to create a Target MCF. Section 13.2.3 then describes how to do the Insert. In this way, a dynamic data granule can be Inserted to the Data Server as if a PGE had produced it.

The Activity Checklist table that follows provides an overview of the process for test data preparation. Column one (**Order**) shows the order in which tasks should be accomplished. Column two (**Role**) lists the Role/Manager/Operator responsible for performing the task. Column three (**Task**) provides a brief explanation of the task. Column four (**Section**) provides the Procedure (P) section number or Instruction (I) section number where details for performing the task can be found. Column five (**Complete?**) is used as a checklist to keep track of which task steps have been completed.

Table 13.2-1. Test Data Preparation - Activity Checklist

Order	Role	Task	Section	Complete?
1	SSI&T	For each ESDT needed by a PGE, generate a source MCF by using the GetMCF Tool.	(I) 13.2.1	
2	SSI&T	For each data granule, create a Target MCF using the above Source MCF. Hand edit the Target MCF to supply metadata that normally would have been supplied by a PGE.	(I) 13.2.2	
3	SSI&T	Insert each dynamic data granule to the Data Server.	(I) 13.2.3	
4	SSI&T	Insert each static data granule to the Data Server.	(I) 13.2.4	

13.2.1 Generating a Metadata Configuration File (Source MCF)

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The SSIT Manager is running.
2. ESDT's are installed onto the Science Data Server.

To Generate the Metadata Configuration File (Source MCF) for the input and output ESDT's, execute the steps that follow.

1 From the SSIT Manager, click on the **Tools** menu, then choose **Data Server** and then **Get MCF**.

- An xterm in which DpAtGetMCF is running will be displayed as SSIT: Acquire MCF..

- Alternatively, the same tool can be invoked by typing at a UNIX prompt on an AIT Sun **DpAtGetMCF.sh**, press **Return**.
- 2 At the program prompt **Configuration Filename (default *defaultConfigFile*)?**
 - Type in **.././cfg/ *defaultConfigFile*** and press **Return**.
 - The ***defaultConfigFile*** will be replaced by the full path name and file name of the default configuration file. The file name will be **DpAtCS_*daac*.CFG** where *daac* will be replaced by one of {GSFC, EDC, LARC, NSIDC}.
 - 3 At the program prompt **ECS mode of operation (enter for default: *defaultMode*)?**, type ***mode***, press **Return** or just press **Return** if the default shown is correct.
 - The ***mode*** refers to the database used and will typically be **TS1**.
 - 4 At the program prompt **ESDT Short Name?**, type ***ESDT ShortName***, press **Return**.
 - The ***ESDTShortName*** is the name of the ESDT that the DpAtGetMCF tool will use to generate the MCF.
 - 5 At the program prompt **ESDT Version?**, type ***ESDTversion***, press **Return** or just press **Return** if the default shown is correct.
 - The ***ESDTversion*** is the version of the ESDT.
 - 6 At the program prompt **Directory to receive MCF (must be full path)?**, type ***MCFpathname***, press **Return**.
 - The ***MCFpathname*** is the full path name to the location where the source MCF will be placed. For example, **/home/jdoe/ssit**.
 - 7 To the final prompt **Hit return to run again, 'q <return> to quit:**, press **Return** to generate another Source MCF or type **q** and press **Return** to quit.
 - If you make a mistake entering any values, press **Return** here; your previous entries are restored as defaults and you won't have to retype them.

Table 13.2.1-2. Generating a Source MCF for an ESDT - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Tools → Data Server → Get MCF	(No action)
2	Configuration Filename	press Return
3	<i>Mode</i>	press Return
4	<i>ESDT ShortName</i>	press Return
5	<i>ESDT Version</i>	press Return
6	<i>Directory to receive MCF</i>	enter the full path of directory to receive MCF, press Return
7	q Return	press Return

13.2.2 Creating a Target MCF (.met) for a Dynamic/Static Granule

A Target MCF file for a corresponding data granule can be created based on the information provided in the Source MCF file and the involved science software package (PGE).

Procedures and tools to create a Target MCF are officially not yet available. In order to create a Target MCF (.met) for a given granule copy an existing .met file and Hand Edit it. Listed below are examples of Source MCF and Target MCF.

Listing 13.2.2-1. Sample Source MCF

```
GROUP = INVENTORYMETADATA
GROUPTYPE = MASTERGROUP
  GROUP = ECSDDataGranule
    OBJECT = LocalGranuleID
      Mandatory = "TRUE"
      Data_Location = "PGE"
      NUM_VAL = 1
      TYPE = "STRING"
    END_OBJECT = LocalGranuleID

OBJECT = SizeMBECSDDataGranule
  Mandatory = "FALSE"
  Data_Location = "DSS"
  NUM_VAL = 1
  TYPE = "DOUBLE"
END_OBJECT = SizeMBECSDDataGranule

OBJECT = ProductionDateTime
  Mandatory = "FALSE"
  Data_Location = "TK"
  NUM_VAL = 1
  TYPE = "DATETIME"
END_OBJECT = ProductionDateTime

OBJECT = DayNightFlag
  Mandatory = "FALSE"
  Data_Location = "PGE"
  NUM_VAL = 1
  TYPE = "STRING"
END_OBJECT = DayNightFlag
OBJECT = ReprocessingActual
  Mandatory = "TRUE"
  Data_Location = "PGE"
  NUM_VAL = 1
  TYPE = "STRING"
END_OBJECT = ReprocessingActual

OBJECT = LocalVersionID
  Mandatory = "TRUE"
  Data_Location = "PGE"
  NUM_VAL = 1
  TYPE = "STRING"
END_OBJECT = LocalVersionID

OBJECT = ReprocessingPlanned
  Mandatory = "TRUE"
  Data_Location = "PGE"
  NUM_VAL = 1
  TYPE = "STRING"
END_OBJECT = ReprocessingPlanned
```

```

END_GROUP = ECSDDataGranule

GROUP = MeasuredParameter
  OBJECT = MeasuredParameterContainer
    Data_Location = "NONE"
    Mandatory = "TRUE"
    CLASS = "M"
  GROUP = QAFlags
    CLASS = "M"
    OBJECT = ScienceQualityFlag
      Mandatory = "FALSE"
      Data_Location = "DP"
      NUM_VAL = 1
      TYPE = "STRING"
    END_OBJECT = ScienceQualityFlag

    OBJECT = AutomaticQualityFlagExplanation
      Mandatory = "TRUE"
      Data_Location = "PGE"
      NUM_VAL = 1
      TYPE = "STRING"
    END_OBJECT = AutomaticQualityFlagExplanation

    OBJECT = AutomaticQualityFlag
      Mandatory = "TRUE"
      Data_Location = "PGE"
      NUM_VAL = 1
      TYPE = "STRING"
    END_OBJECT = AutomaticQualityFlag

    OBJECT = OperationalQualityFlagExplanation
      Mandatory = "FALSE"
      Data_Location = "DAAC"
      NUM_VAL = 1
      TYPE = "STRING"
    END_OBJECT = OperationalQualityFlagExplanation

    OBJECT = OperationalQualityFlag
      Mandatory = "FALSE"
      Data_Location = "DAAC"
      NUM_VAL = 1
      TYPE = "STRING"
    END_OBJECT = OperationalQualityFlag

    OBJECT = ScienceQualityFlagExplanation
      Mandatory = "FALSE"
      Data_Location = "DP"
      NUM_VAL = 1
      TYPE = "STRING"
    END_OBJECT = ScienceQualityFlagExplanation
  END_GROUP = QAFlags

GROUP = QAStats
  CLASS = "M"
  OBJECT = QAPercentMissingData
    Mandatory = "TRUE"
    Data_Location = "PGE"
    NUM_VAL = 1

```

```

        TYPE = "INTEGER"
        END_OBJECT = QAPercentMissingData
    END_GROUP = QASStats

    OBJECT = ParameterName
        Mandatory = "TRUE"
        CLASS = "M"
        Data_Location = "PGE"
        NUM_VAL = 1
        TYPE = "STRING"
    END_OBJECT = ParameterName
    END_OBJECT = MeasuredParameterContainer
    END_GROUP = MeasuredParameter

GROUP = OrbitCalculatedSpatialDomain
    OBJECT = OrbitCalculatedSpatialDomainContainer
        Data_Location = "NONE"
        Mandatory = "TRUE"
        CLASS = "M"
    OBJECT = EquatorCrossingDate
        Mandatory = "TRUE"
        CLASS = "M"
        Data_Location = "PGE"
        NUM_VAL = 1
        TYPE = "DATE"
    END_OBJECT = EquatorCrossingDate

    OBJECT = EquatorCrossingTime
        Mandatory = "TRUE"
        CLASS = "M"
        Data_Location = "PGE"
        NUM_VAL = 1
        TYPE = "TIME"
    END_OBJECT = EquatorCrossingTime

    OBJECT = OrbitNumber
        Mandatory = "TRUE"
        CLASS = "M"
        Data_Location = "PGE"
        NUM_VAL = 1
        TYPE = "INTEGER"
    END_OBJECT = OrbitNumber

    OBJECT = EquatorCrossingLongitude
        Mandatory = "TRUE"
        CLASS = "M"
        Data_Location = "PGE"
        NUM_VAL = 1
        TYPE = "DOUBLE"
    END_OBJECT = EquatorCrossingLongitude
    END_OBJECT = OrbitCalculatedSpatialDomainContainer
    END_GROUP = OrbitCalculatedSpatialDomain

GROUP = CollectionDescriptionClass
    OBJECT = VersionID
        Mandatory = "TRUE"
        Data_Location = "MCF"
        NUM_VAL = 1

```



```

    TYPE = "STRING"
    Value = "2"
END_OBJECT = VersionID

OBJECT = ShortName
    Mandatory = "TRUE"
    Data_Location = "MCF"
    NUM_VAL = 1
    TYPE = "STRING"
    Value = "MOD35_L2"
END_OBJECT = ShortName
END_GROUP = CollectionDescriptionClass

GROUP = InputGranule
    OBJECT = InputPointer
        Mandatory = "TRUE"
        Data_Location = "PGE"
        NUM_VAL = 25
        TYPE = "STRING"
    END_OBJECT = InputPointer
END_GROUP = InputGranule
GROUP = SpatialDomainContainer
    GROUP = HorizontalSpatialDomainContainer
        GROUP = BoundingBoxRectangle
            OBJECT = EastBoundingCoordinate
                Mandatory = "TRUE"
                Data_Location = "PGE"
                NUM_VAL = 1
                TYPE = "DOUBLE"
            END_OBJECT = EastBoundingCoordinate
            OBJECT = WestBoundingCoordinate
                Mandatory = "TRUE"
                Data_Location = "PGE"
                NUM_VAL = 1
                TYPE = "DOUBLE"
            END_OBJECT = WestBoundingCoordinate

            OBJECT = SouthBoundingCoordinate
                Mandatory = "TRUE"
                Data_Location = "PGE"
                NUM_VAL = 1
                TYPE = "DOUBLE"
            END_OBJECT = SouthBoundingCoordinate

            OBJECT = NorthBoundingCoordinate
                Mandatory = "TRUE"
                Data_Location = "PGE"
                NUM_VAL = 1
                TYPE = "DOUBLE"
            END_OBJECT = NorthBoundingCoordinate
        END_GROUP = BoundingBoxRectangle
    END_GROUP = HorizontalSpatialDomainContainer
END_GROUP = SpatialDomainContainer

GROUP = RangeDateTime
    OBJECT = RangeEndingDate
        Mandatory = "TRUE"
        Data_Location = "PGE"

```

```

    NUM_VAL = 1
    TYPE = "DATE"
END_OBJECT = RangeEndingDate

OBJECT = RangeEndingTime
    Mandatory = "TRUE"
    Data_Location = "PGE"
    NUM_VAL = 1
    TYPE = "TIME"
END_OBJECT = RangeEndingTime

OBJECT = RangeBeginningDate
    Mandatory = "TRUE"
    Data_Location = "PGE"
    NUM_VAL = 1
    TYPE = "DATE"
END_OBJECT = RangeBeginningDate
OBJECT = RangeBeginningTime
    Mandatory = "TRUE"
    Data_Location = "PGE"
    NUM_VAL = 1
    TYPE = "TIME"
END_OBJECT = RangeBeginningTime
END_GROUP = RangeDateTime

GROUP = PGEVersionClass
    OBJECT = PGEVersion
        Mandatory = "TRUE"
        Data_Location = "PGE"
        NUM_VAL = 1
        TYPE = "STRING"
    END_OBJECT = PGEVersion
END_GROUP = PGEVersionClass
GROUP = AncillaryInputGranule
    OBJECT = AncillaryInputGranuleContainer
        Data_Location = "NONE"
        Mandatory = "TRUE"
        CLASS = "M"
    OBJECT = AncillaryInputPointer
        Mandatory = "TRUE"
        CLASS = "M"
        Data_Location = "PGE"
        NUM_VAL = 1
        TYPE = "STRING"
    END_OBJECT = AncillaryInputPointer
    OBJECT = AncillaryInputType
        Mandatory = "TRUE"
        CLASS = "M"
        Data_Location = "PGE"
        NUM_VAL = 1
        TYPE = "STRING"
    END_OBJECT = AncillaryInputType
    END_OBJECT = AncillaryInputGranuleContainer
END_GROUP = AncillaryInputGranule
GROUP = AdditionalAttributes
    OBJECT = AdditionalAttributesContainer
        Data_Location = "NONE"
        Mandatory = "FALSE"

```

```

CLASS = "M"
OBJECT = AdditionalAttributeName
Mandatory = "FALSE"
CLASS = "M"
Data_Location = "PGE"
NUM_VAL = 1
TYPE = "STRING"
END_OBJECT = AdditionalAttributeName
GROUP = InformationContent
CLASS = "M"
OBJECT = ParameterValue
Mandatory = "FALSE"
Data_Location = "PGE"
NUM_VAL = 1
TYPE = "FLOAT"
END_OBJECT = ParameterValue
END_GROUP = InformationContent
END_OBJECT = AdditionalAttributesContainer
END_GROUP = AdditionalAttributes
GROUP = AssociatedPlatformInstrumentSensor
OBJECT = AssociatedPlatformInstrumentSensorContainer
Data_Location = "NONE"
Mandatory = "TRUE"
CLASS = "M"
OBJECT = AssociatedSensorShortName
Mandatory = "FALSE"
CLASS = "M"
Data_Location = "PGE"
NUM_VAL = 1
TYPE = "STRING"
END_OBJECT = AssociatedSensorShortName
OBJECT = AssociatedPlatformShortName
Mandatory = "FALSE"
CLASS = "M"
Data_Location = "PGE"
NUM_VAL = 1
TYPE = "STRING"
END_OBJECT = AssociatedPlatformShortName
OBJECT = AssociatedInstrumentShortName
Mandatory = "FALSE"
CLASS = "M"
Data_Location = "PGE"
NUM_VAL = 1
TYPE = "STRING"
END_OBJECT = AssociatedInstrumentShortName
END_OBJECT = AssociatedPlatformInstrumentSensorContainer
END_GROUP = AssociatedPlatformInstrumentSensor
END_GROUP = INVENTORYMETADATA
GROUP = ARCHIVEDMETADATA
GROUPTYPE = MASTERGROUP
GROUP = AlgorithmPackage
OBJECT = AlgorithmPackageAcceptanceDate
Data_Location = "PGE"
Mandatory = "TRUE"
TYPE = "STRING"
NUM_VAL = 1
END_OBJECT = AlgorithmPackageAcceptanceDate
OBJECT = AlgorithmPackageMaturityCode

```

```

Data_Location = "PGE"
Mandatory = "TRUE"
TYPE = "STRING"
NUM_VAL = 1
END_OBJECT = AlgorithmPackageMaturityCode
OBJECT = AlgorithmPackageName
Data_Location = "PGE"
Mandatory = "TRUE"
TYPE = "STRING"
NUM_VAL = 1
END_OBJECT = AlgorithmPackageName
OBJECT = AlgorithmPackageVersion
Data_Location = "PGE"
Mandatory = "TRUE"
TYPE = "STRING"
NUM_VAL = 1
END_OBJECT = AlgorithmPackageVersion
OBJECT = LocalInputGranuleID
Data_Location = "PGE"
Mandatory = "TRUE"
TYPE = "STRING"
NUM_VAL = 10
END_OBJECT = LocalInputGranuleID
OBJECT = InstrumentName
Data_Location = "PGE"
Mandatory = "TRUE"
TYPE = "STRING"
NUM_VAL = 1
END_OBJECT = InstrumentName
OBJECT = LongName
Data_Location = "PGE"
Mandatory = "TRUE"
TYPE = "STRING"
NUM_VAL = 1
END_OBJECT = LongName
END_GROUP = AlgorithmPackage

GROUP = GPolygon
OBJECT = GPolygonContainer
Data_Location = "NONE"
Mandatory = "TRUE"
CLASS = "M"
GROUP = GRing
CLASS = "M"
OBJECT = ExclusionGRingFlag
Mandatory = "TRUE"
Data_Location = "PGE"
NUM_VAL = 1
TYPE = "STRING"
END_OBJECT = ExclusionGRingFlag
END_GROUP = GRing
GROUP = GRingPoint
CLASS = "M"
OBJECT = GRingPointLongitude
Mandatory = "TRUE"
Data_Location = "PGE"
NUM_VAL = 4
TYPE = "DOUBLE"

```

```

END_OBJECT = GRingPointLongitude
OBJECT = GRingPointLatitude
Mandatory = "TRUE"
Data_Location = "PGE"
NUM_VAL = 4
TYPE = "DOUBLE"
END_OBJECT = GRingPointLatitude
OBJECT = GRingPointSequenceNo
Mandatory = "TRUE"
Data_Location = "PGE"
NUM_VAL = 4
TYPE = "INTEGER"
END_OBJECT = GRingPointSequenceNo
END_GROUP = GRingPoint
END_OBJECT = GPolygonContainer
END_GROUP = GPolygon
OBJECT = Cloud_Mask_Algorithm_Version_Number
Data_Location = "PGE"
Mandatory = "TRUE"
TYPE = "STRING"
NUM_VAL = 1
END_OBJECT = Cloud_Mask_Algorithm_Version_Number
END_GROUP = ARCHIVEDMETADATA
END

```

Listing 13.2.2-2. Sample Corresponding Target MCF

GROUP	= INVENTORYMETADATA
GROUPTYPE	= MASTERGROUP
GROUP	= ECSDATAGRANULE
OBJECT	= LOCALGRANULEID
NUM_VAL	= 1
VALUE	= "LocalGranuleID"
END_OBJECT	= LOCALGRANULEID
OBJECT	= SIZEMBECSDATAGRANULE
NUM_VAL	= 1
VALUE	= 30.000000
END_OBJECT	= SIZEMBECSDATAGRANULE
OBJECT	= PRODUCTIONDATETIME
NUM_VAL	= 1
VALUE	= "1997-11-26T18:40:31.000Z"
END_OBJECT	= PRODUCTIONDATETIME
OBJECT	= DAYNIGHTFLAG
NUM_VAL	= 1
VALUE	= "Day"
END_OBJECT	= DAYNIGHTFLAG
OBJECT	= REPROCESSINGACTUAL
NUM_VAL	= 1
VALUE	= "reprocessed once"
END_OBJECT	= REPROCESSINGACTUAL
OBJECT	= LOCALVERSIONID
NUM_VAL	= 1
VALUE	= "LocalVersionID"
END_OBJECT	= LOCALVERSIONID
OBJECT	= REPROCESSINGPLANNED
NUM_VAL	= 1
VALUE	= "no further update anticipated"
END_OBJECT	= REPROCESSINGPLANNED
END_GROUP	= ECSDATAGRANULE
GROUP	= MEASUREDPARAMETER
OBJECT	= MEASUREDPARAMETERCONTAINER
CLASS	= "1"
GROUP	= QAFLAGS
CLASS	= "1"
OBJECT	= SCIENCEQUALITYFLAG
NUM_VAL	= 1
CLASS	= "1"
VALUE	= "Passed"
END_OBJECT	= SCIENCEQUALITYFLAG

OBJECT	= AUTOMATICQUALITYFLAGEXPLANATION
NUM_VAL	= 1
CLASS	= "1"
VALUE	= "AutomaticQualityFlagExplanation"
END_OBJECT	= AUTOMATICQUALITYFLAGEXPLANATION
OBJECT	= AUTOMATICQUALITYFLAG
NUM_VAL	= 1
CLASS	= "1"
VALUE	= "Passed"
END_OBJECT	= AUTOMATICQUALITYFLAG
OBJECT	= OPERATIONALQUALITYFLAGEXPLANATION
NUM_VAL	= 1
CLASS	= "1"
VALUE	= "OperationalQualityFlagExplanation"
END_OBJECT	= OPERATIONALQUALITYFLAGEXPLANATION
OBJECT	= OPERATIONALQUALITYFLAG
NUM_VAL	= 1
CLASS	= "1"
VALUE	= "Passed"
END_OBJECT	= OPERATIONALQUALITYFLAG
OBJECT	= SCIENCEQUALITYFLAGEXPLANATION
NUM_VAL	= 1
CLASS	= "1"
VALUE	= "ScienceQualityFlagExplanation"
END_OBJECT	= SCIENCEQUALITYFLAGEXPLANATION
END_GROUP	= QAFLAGS
GROUP	= QASTATS
CLASS	= "1"
OBJECT	= QAPERCENTMISSINGDATA
NUM_VAL	= 1
CLASS	= "1"
VALUE	= 0
END_OBJECT	= QAPERCENTMISSINGDATA
OBJECT	= QAPERCENTOUTOFBOUNDSDATA
NUM_VAL	= 1
CLASS	= "1"
VALUE	= 0
END_OBJECT	= QAPERCENTOUTOFBOUNDSDATA
OBJECT	= QAPERCENTCLOUDCOVER
NUM_VAL	= 1
CLASS	= "1"
VALUE	= 50
END_OBJECT	= QAPERCENTCLOUDCOVER
OBJECT	= QAPERCENTINTERPOLATEDDATA
NUM_VAL	= 1
CLASS	= "1"
VALUE	= 100
END_OBJECT	= QAPERCENTINTERPOLATEDDATA

END_GROUP	= QASTATS
OBJECT	= PARAMETERNAME
CLASS	= "1"
NUM_VAL	= 1
VALUE	= "ParameterName"
END_OBJECT	= PARAMETERNAME
END_OBJECT	= MEASUREDPARAMETERCONTAINER
END_GROUP	= MEASUREDPARAMETER
GROUP	= ORBITCALCULATEDSPATIALDOMAIN
OBJECT	= ORBITCALCULATEDSPATIALDOMAINCONTAINER
CLASS	= "1"
OBJECT	= ORBITALMODELNAME
CLASS	= "1"
NUM_VAL	= 1
VALUE	= "OrbitalModelName"
END_OBJECT	= ORBITALMODELNAME
OBJECT	= STARTORBITNUMBER
CLASS	= "1"
NUM_VAL	= 1
VALUE	= 1
END_OBJECT	= STARTORBITNUMBER
OBJECT	= EQUATORCROSSINGDATE
CLASS	= "1"
NUM_VAL	= 1
VALUE	= "1993-01-01"
END_OBJECT	= EQUATORCROSSINGDATE
OBJECT	= EQUATORCROSSINGTIME
CLASS	= "1"
NUM_VAL	= 1
VALUE	= "01:00:00.000000Z"
END_OBJECT	= EQUATORCROSSINGTIME
OBJECT	= ORBITNUMBER
CLASS	= "1"
NUM_VAL	= 1
VALUE	= 1
END_OBJECT	= ORBITNUMBER
OBJECT	= EQUATORCROSSINGLONGITUDE
CLASS	= "1"
NUM_VAL	= 1
VALUE	= 90.000000
END_OBJECT	= EQUATORCROSSINGLONGITUDE
OBJECT	= STOPORBITNUMBER
CLASS	= "1"
NUM_VAL	= 1
VALUE	= 1


```

END_OBJECT      = STOPORBITNUMBER

END_OBJECT      = ORBITCALCULATEDSPATIALDOMAINCONTAINER

END_GROUP       = ORBITCALCULATEDSPATIALDOMAIN

GROUP           = COLLECTIONDESCRIPTIONCLASS

OBJECT          = VERSIONID
  NUM_VAL       = 1
  VALUE         = "2.0"
END_OBJECT      = VERSIONID

OBJECT          = SHORTNAME
  NUM_VAL       = 1
  VALUE         = "MOD35_L2"
END_OBJECT      = SHORTNAME

END_GROUP       = COLLECTIONDESCRIPTIONCLASS

GROUP           = INPUTGRANULE

OBJECT          = INPUTPOINTER
  NUM_VAL       = 1
  VALUE         = "InputPointer"
END_OBJECT      = INPUTPOINTER

END_GROUP       = INPUTGRANULE

GROUP           = SPATIALDOMAINCONTAINER

GROUP           = HORIZONTALSPATIALDOMAINCONTAINER

GROUP           = GPOLYGON

OBJECT          = GPOLYGONCONTAINER
  CLASS         = "1"

GROUP           = GRINGPOINT
  CLASS         = "1"

OBJECT          = GRINGPOINTLONGITUDE
  CLASS         = "1"
  NUM_VAL       = 1
  VALUE         = 180.000000
END_OBJECT      = GRINGPOINTLONGITUDE

OBJECT          = GRINGPOINTLATITUDE
  CLASS         = "1"
  NUM_VAL       = 1
  VALUE         = 90.000000
END_OBJECT      = GRINGPOINTLATITUDE

OBJECT          = GRINGPOINTSEQUENCENO
  CLASS         = "1"
  NUM_VAL       = 1
  VALUE         = 1
END_OBJECT      = GRINGPOINTSEQUENCENO

```

END_GROUP	= GRINGPOINT
GROUP	= GRING
CLASS	= "1"
OBJECT	= EXCLUSIONGRINGFLAG
NUM_VAL	= 1
CLASS	= "1"
VALUE	= "Y"
END_OBJECT	= EXCLUSIONGRINGFLAG
END_GROUP	= GRING
END_OBJECT	= GPOLYGONCONTAINER
END_GROUP	= GPOLYGON
GROUP	= ZONEIDENTIFIERCLASS
OBJECT	= ZONEIDENTIFIER
NUM_VAL	= 1
VALUE	= "State Plane Coordinate System of 1983"
END_OBJECT	= ZONEIDENTIFIER
END_GROUP	= ZONEIDENTIFIERCLASS
END_GROUP	= HORIZONTALSPATIALDOMAINCONTAINER
END_GROUP	= SPATIALDOMAINCONTAINER
GROUP	= RANGEDATETIME
OBJECT	= RANGEENDINGDATE
NUM_VAL	= 1
VALUE	= "1996-08-05"
END_OBJECT	= RANGEENDINGDATE
OBJECT	= RANGEENDINGTIME
NUM_VAL	= 1
VALUE	= "15:59:00.000000Z"
END_OBJECT	= RANGEENDINGTIME
OBJECT	= RANGEBEGINNINGDATE
NUM_VAL	= 1
VALUE	= "1996-08-05"
END_OBJECT	= RANGEBEGINNINGDATE
OBJECT	= RANGEBEGINNINGTIME
NUM_VAL	= 1
VALUE	= "15:54:00.000000Z"
END_OBJECT	= RANGEBEGINNINGTIME
END_GROUP	= RANGEDATETIME
GROUP	= PGEVERSIONCLASS
OBJECT	= PGEVERSION

```

    NUM_VAL      = 1
    VALUE        = "1"
    END_OBJECT   = PGEVERSION

END_GROUP      = PGEVERSIONCLASS

GROUP          = ANCILLARYINPUTGRANULE

OBJECT
  CLASS        = ANCILLARYINPUTGRANULECONTAINER
  CLASS        = "1"

OBJECT
  CLASS        = ANCILLARYINPUTPOINTER
  CLASS        = "1"
  NUM_VAL      = 1
  VALUE        = "AncillaryInputPointer"
  END_OBJECT   = ANCILLARYINPUTPOINTER

OBJECT
  CLASS        = ANCILLARYINPUTTYPE
  CLASS        = "1"
  NUM_VAL      = 1
  VALUE        = "Geolocation"
  END_OBJECT   = ANCILLARYINPUTTYPE

END_OBJECT     = ANCILLARYINPUTGRANULECONTAINER

END_GROUP      = ANCILLARYINPUTGRANULE

END_GROUP      = INVENTORYMETADATA

END

```

13.2.3 Inserting Dynamic Data Granules to the Data Server

In order for dynamic data files to be used both during the SSI&T and in production, this file must exist in the Data Server and be accessible by the local machine. A program called the Insert Test Dynamic File can be used for Inserting a dynamic data granule into the Data Server.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ESDT descriptor file for the granule to be Inserted exists in the ODL format on the Data Server.
2. The Target MCF for this data granule has been created for the Insert.

To Insert a dynamic granule to the Data Server, execute the following steps:

- 1 From the SSIT Manager, click on the **Tools** menu, then choose **Data Server** and then **Insert Test Dynamic**.
 - An xterm in which DpAtInsertTestFile.sh is running will be displayed.
 - Alternatively, the same tool can be invoked by typing at a UNIX prompt on an AIT Sun **DpAtInsertTestFile.sh**, press **Return**.
 - The DpAtInsertTestFile.sh program will prompt for further information.

- 2 At the program prompt **Configuration filename (default *defaultConfigFile*)?**, press **Return**.
 - The default configuration file for this tool will be used.
 - The *defaultConfigFile* will be replaced by the full path name and file name of the default configuration file. The file name will be *DpAtID_daac.CFG* where *daac* will be replaced by one of {GSFC, EDC, LARC, NSIDC}.
- 3 At the program prompt **ECS Mode of operation? (enter for default: OPS)**
 - If **OPS** is not the correct mode, then enter in the correct mode (e.g. *ssit*) and press **Return**.
- 4 At the program prompt **ESDT name?** type *ESDTShortName*, press **Return**
 - The *ESDTShortName* is the ShortName of the ESDT descriptor file corresponding to this granule to be Inserted. For example, type **CER01T**, press **Return**.
- 5 At the program prompt **ESDT Version (enter for default: 1)** press **Return**.
- 6 At the program prompt **Staged Filename to Insert? (including FULL path)** type *pathname/GranuleFilename*, press **Return**
 - The *pathname/GranuleFilename* is the full path name and file name of the data granule to be Inserted. For example, type */data/CERES/CER01T.MCF*, press **Return**.
- 7 At the program prompt **Associated ASCII Metadata Filename to Insert (including FULL path) (default *pathname/GranuleFileName.met*)?**, press **Return**.
 - The default is the file name of the granule to insert with the .met file name extension. If the default is not correct, then the file name of this file must be entered. For example, */data/CERES/CER01T.met*.
 - The dynamic data granule will be Inserted to the Data Server. For reference, the Data Server Universal Reference (UR) will be printed on the screen.
- 8 At the program prompt **Hit return to run again, 'q <return>' to quit:** type **q** and press **Return** to quit or just press **Return** to insert additional dynamic granules.
 - If continuing, repeat steps 2 through 7.

Table 13.2.3-1. Inserting Dynamic Data Granules to the Data Server - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Tools → Data Server → Insert Test Dynamic	(No action)
2	<i>Configuration FileName</i>	press Return
3	<i>Mode</i>	press Return
4	<i>ESDTShortName</i>	press Return
5	<i>ESDTVersion</i>	press Return
6	<i>pathname/GranuleFilename</i>	press Return
7	<i>pathname/GranuleFileName.met</i>	press Return
6	q Return	press Return

13.2.4 Inserting Static Data Granules to the Data Server

In order for static data files to be used both during the SSI&T and in production, this file must exist in the Data Server and be accessible by the local machine. A program called the Insert Static File can be used for Inserting a static data granule into the Data Server. Note that Source MCFs are static granules which must be inserted in this way.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The SSIT Manager is running.
2. The ESDT descriptor file for the granule to be Inserted exists in the ODL format on the Data Server.
3. The Target MCF for this data granule has been created for the Insert.

To Insert the static granule data file to the Data Server, execute the steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **D**ata **S**erver and then **I**nsert **S**tatic.
 - An xterm in which DpAtInsertStaticFile.sh is running will be displayed.
 - Alternatively, the same tool can be invoked by typing at a UNIX prompt on an AIT Sun **DpAtInsertStaticFile.sh**, press **Return**.
 - The DpAtInsertStaticFile.sh program will prompt for further information.
- 2 At the program prompt **Configuration filename (default defaultConfigFile)?**
 - Type **../cf/defaultConfigFile** and press **Return**.
 - The **defaultConfigFile** will be replaced by the full path name and file name of the default configuration file. The file name will be DpAtIS_*daac*.CFG where *daac* will be replaced by one of {GSFC, EDC, LARC, NSIDC}.
- 3 At the program prompt **ECS mode of operation (default OPS)?**, press **Return**
 - If **OPS** is not the correct mode, then enter in the correct mode (*e.g.* ssit) and press **Return**.
- 4 At the program prompt **ESDT Short Name?** type **ESDTShortName**, press **Return**
 - The **ESDTShortName** is the ShortName of the ESDT descriptor file corresponding to this granule to be Inserted. For example, type **CADMERLW**, press **Return**.
- 5 At the program prompt **ESDT Version?** type the version number and press **Return**.
- 6 At the program prompt **Science Group for Static file (one of {C, L, D, O} followed by a 3 digit number)?**, type **ScienceGroupID**, press **Return**
 - The **ScienceGroupID** is an identifier used to define the file type as a coefficient file, a lookup table file, or a MCF. It distinguishes static granules of different types

which share the same ESDT. For instance, for a coefficient file, use **Cn**, where number *n* could be 0, 1, 2...; this number *n* needs to be matched with the number *n* in the PGE_PGENAME#Version.odl file (see Section 13.1.2). For example, type **C001**, press **Return**.

- The Science Group ID must match what was edited into the PGE metadata ODL file for that PCF entry (Section 13.1.2).

- 7 At the program prompt **Filename to insert?**, type *pathname/GranuleFileName*, press **Return**
 - The *pathname/GranuleFileName* is the full path name and file name of the static data granule to be Inserted. For example, type */data/CERES/CADMERLW.MCF*, press **Return**.
- 8 At the program prompt **Associated ASCII Filename to Insert (default *pathname/GranuleFileName.met*)?**, press **Return**.
 - The default is the file name of the granule to insert with the .met file name extension. If the default is not correct, then the file name of this file must be entered.
 - The static data granule will be Inserted to the Data Server.
- 9 At the program prompt **Hit return to run again, 'q <return>' to quit:** type **q** and press **Return** to quit or just press **Return** to insert additional dynamic granules.
 - If continuing, repeat steps 2 through 8.

Table 13.2.4-1. Inserting Static Data Granules to the Data Server - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>D</u> ata <u>S</u> erver → <u>I</u> nsert Static	(No action)
2	<i>Configuration FileName</i>	press Return
3	<i>Mode</i>	press Return
4	<i>ESDTShortName</i>	press Return
5	<i>ESDT Version</i>	press Return
6	<i>ScienceGroupID</i>	press Return
7	<i>pathname/GranuleFileName</i>	press Return
8	<i>pathname/GranuleFileName.met</i>	press Return
9	q Return	press Return

14. PGE Planning, Processing, and Product Retrieval

14.1 Using the Production Request Editor

When standalone tests (Run from the command line) have completed successfully and information about the PGE has entered into the PDPS Database (through PGE registration and file population), the PGE is ready to be run through the automated ECS PDPS environment.

To process Science data, a Production Request (PR) must be submitted to the ECS system. The Production Request Editor GUI accomplishes this function. Only one PR may be submitted at a time. A single PR is exploded by the PDPS into one or more jobs called Data Processing Requests (DPRs). The number of DPRs that are created for a single PR is determined by the number needed to cover the requested time interval. Some PRs may only require one DPR.

14.1.1 Invoking the Production Request Editor

Currently, the Production Request Editor is invoked from a command line script. In the future, this will be done by clicking on the icon for the PR Editor on the ECS Desktop. Once the Production Request Editor is invoked, it brings up a screen with five tabs at the top for selection (as shown in Fig. 14.1.1-1). The first tab is labeled “Planning”. Selection of this tab displays a list of four capabilities available for the PR Editor by selecting the other tabs at the top of the primary GUI screen: PR Edit, PR List, DPR View, and DPR List.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The PGE and all associated files have been registered in the PDPS Database.
2. The PGE has been successfully compiled and linked with the DAAC version of the SDP Toolkit.
3. The required servers are up and running.

To invoke the Production Request Editor GUI, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then telnet to a PLN host and change to the proper directory.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the PLN host.
 - Set the DISPLAY environment variable

- cd to the directory where the scripts are located. (e.g. /usr/ecs/TS1/CUSTOM/utilities).
- 2 At the UNIX prompt on the PLN host (e.g. illiad), type **EcPlStartPRE *mode application_id* &**, press **Return**.
- The ***mode*** is the operations mode.(e.g. TS1)
 - The ***application_id*** is a numerical number. (e.g. 1)
 - For example, type **EcPlStartPRE TS1 1 &**, press **Return**.
 - Various messages from the Production Request Editor may appear in this window as it is running. For this reason, avoid using this window for other tasks until the Production Request Editor has terminated.
- 3 In the Production Request Editor, click on one of the tabs PR Edit, PR List, DPR View, or DPR List corresponding to desired task.
- To define a new Production Request or edit a Production Request, click on PR Edit. Proceed to Section 14.1.2.
 - To review or list a Production Request, click on PR List.
 - To view or inspect a Data Processing Request, click on View.
 - To review or inspect a Data Processing Request, click on List.
- 4 When tasks are completed in the Production Request Editor GUI, click on the **File** menu, then choose **Exit**.
- The Production Request Editor will disappear.

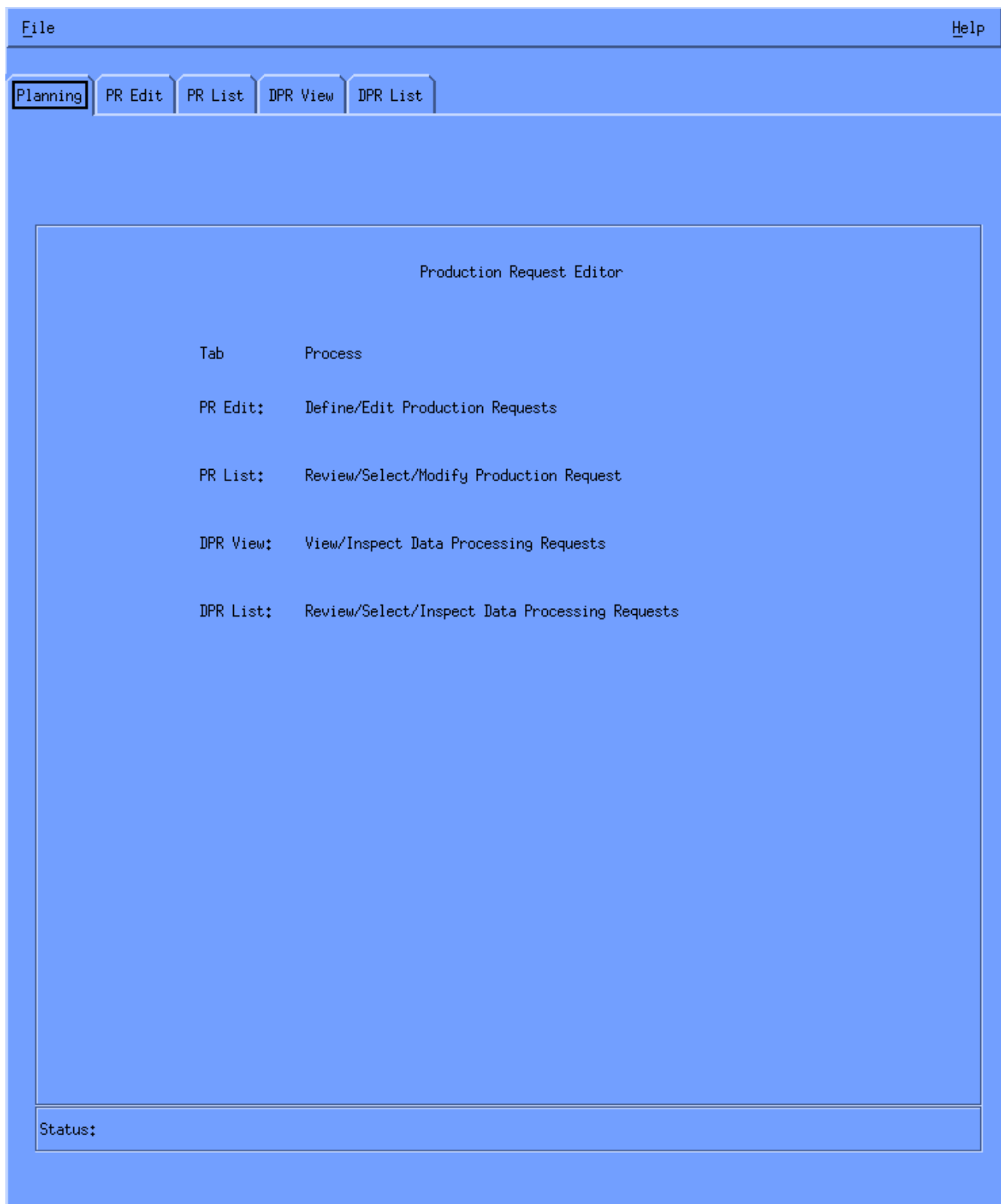


Figure 14.1.1-1. Production Request Editor (Planning) GUI.

Table 14.1.1-1. Invoking the Production Request Editor - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	T ools → X term	telnet to PLN host
2	E cPIStartPRE <i>mode application_id</i> &	press Return
3	(No entry)	choose a Production Request activity
4	F ile → E xit	(No action)

14.1.2 Defining a New Production Request

A Production Request (PR) is a request for data production of granules between a start date/time and an end date/time. A PR will explode into one or more Data Processing Requests (DPR) depending upon the time interval involved. Each DPR corresponds to the execution of a PGE. Therefore, a PR results in the execution of a PGE one or more times. Only one PGE is involved in a single Production Request.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The user has selected the **PR Edit** tab from the Production Request Editor (Section 14.1.1).
2. The PGE involved in the Production Request has been registered in the PDPS database (Section 13.1).

To define a new Production Request, execute the procedure steps that follow:

- 1 From the Production Request Editor GUI, click on the **PR Edit** tab.
 - The PR Edit page will be displayed as shown in **Fig. 14.1.2-1**.
- 2 In the field labeled **PR Name:**, enter **New** or verify that **New** is already entered as the default.
- 3 The PGE for the Production Request must be selected from a list. To do this, click on the **PGE...** button.
 - A GUI labeled **PGE Selection** will be displayed within which registered PGEs will be listed. The appropriate PGE can then be selected by clicking on it and then on the **OK** button.
 - The selected PGE will then be used to populate **Satellite Name**, **Instrument Name**, **PGE Name**, and **PGE Version** fields of the main GUI.
- 4 In the field labeled **Priority:**, enter *priority*.

- The *priority* is the priority to be assigned to this Production Request in the range 0 through 99 with 0 being the lowest priority and 99 the highest. For example, enter **40**.
- 5 Optionally, to view and possibly override PGE runtime parameter settings, click on the **PGE Parameters...** button.
 - A GUI labeled **PGE Parameter Mappings** will be displayed.
 - Parameter names, logical IDs (from the PCF), and default settings will be listed. To override a setting, click on a parameter name. In the field labeled **Parameter Mapping** enter a new value and press **Return**. The value entered will show up in the **Value Override** column.
 - Other parameters may be changed in the same way.
 - When complete, click on the **OK** button. The **PGE Parameter Mappings** GUI will disappear.
 - 6 In the Production Request Editor GUI, select a **Duration** option. Two options are provided:
 - **UTC Time** for time range.
 - **Orbit** for orbit number range.
 - 7 In the Production Request Editor GUI, enter *StartDate* and *StartTime* in fields labeled **Begin**, respectively.
 - The *StartDate* and *StartTime* are the start date and time of the Production Request and should be entered in the mm/dd/yy and hh:mm:ss formats.
 - 8 In the case of UTC Time duration, enter *EndDate* and *EndTime* in fields labeled **End**, respectively.
 - The *Enddate* and *EndTime* are the end date and time of the Production Request and should be entered in the mm/dd/yy and hh:mm:ss formats.
 - 9 Optionally, enter *Comment* in field labeled **Comment:**.
 - This comment will be displayed whenever this Production Request is brought up and viewed.
 - 10 When Production Request is complete, click on **File** menu and select **Save As....**
 - A GUI labeled **File Selection** will be displayed.
 - In the field labeled **Selection**, enter a user-defined name to be assigned to the Production Request. Then click on the **OK** button. A message box will be displayed stating “Production Request Explosion into DPRs ok, *n* DPRs Generated”, where *n* will be the number of DPRs. Click on the **Ok** button. A second message box stating “Write to Database of Production Request ok”; again click **Ok**.
 - Note that you will not be allowed to enter a PR name that already exists. PR names that already exist will be displayed in the main window.
 - The Production Request will then be saved under the name specified.

File
Edit
Help

Planning
PR Edit
PR List
DPR View
DPR List

Production Request Identification

PR Name: New

Origination Date: 12/15/97 11:34:32

PR Type: Routine

Originator:

Priority: 0

Request Definition

Satellite Name:

Instrument Name:

PGE Name:

PGE Version:

Profile Id: 0

PGE ...

PGE Parameters...

Metadata Checks...

Alternate Input Values..

Duration
UTC Time
Orbit

Begin 12 / 15 / 1997 - 16 : 34 : 32

End 12 / 15 / 1997 - 16 : 34 : 32

Tile Id 0

From 0

To 0

Intermittent DPR

Skip 0

Keep 0

SkipFirst

Comment:

Status:

Figure 14.1.2-1. PR Editor GUI.

14-6

162-TD-001-004

Table 14.1.2-1. Defining a New Production Request - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	PR Edit	click tab
2	New	verify or enter
3	PGE...	click button then select a PGE
4	<i>priority</i>	enter
5	(Optional) PGE Parameters...	click button then set a parameter value
6	<i>Select Duration Option</i>	Click one of the choice
7	<i>StartDate</i> <i>StartTime</i>	enter enter
8	<i>EndDate</i> <i>EndTime</i>	enter enter
9	(Optional) <i>Comment</i>	click parameter
10	<u>F</u> ile → <u>S</u> ave <u>A</u> s, then PR name	enter

14.1.3 Viewing Production Requests

A Production Request (PR) is a request for data production of granules between a start date/time and an end date/time. A PR will explode into one or more Data Processing Requests (DPR) depending upon the time interval involved. Each DPR corresponds to the execution of a PGE. Therefore, a PR results in the execution of a PGE one or more times. Only one PGE is involved in a single Production Request.

This procedure describes how to view PRs that have already been defined (see Section 14.1.2). It assumes that the **PR List** tab has been selected from the Production Request Editor.

The information listed for each PR is:

- PR Name - The name assigned to the Production Request when it was defined.
- PGE ID - The name of the PGE involved in the PR.
- Priority - The priority (0 - 99) of the PR assigned when it was defined.
- Start - The start date and time of the PR.
- End - The end date and time of the PR.
- Comment - Any comment that was entered when the PR was defined.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The user has selected the **PR List** tab from the Production Request Editor (Section 14.1.1).

To view Production Requests, execute the procedure steps that follow:

- 1 From the Production Request Editor GUI, click on the **PR List** tab.
 - The PR List page will be displayed as shown in **Fig. 14.1.3-1**.
- 2 View the listed PRs. Optionally, find a PR by entering a search string in the field next to the **Find** button and then clicking on the **Find** button.
- 3 To modify a PR listed, click on the PR in the list and from the **File** menu select **Save As...**
 - In the **File Selection** GUI, replace the current PR name shown in the **Selection** field with a new PR name. Then click on the **OK** button.
 - When modifying an existing PR, it must be saved under a new PR name.
 - Next, click on the **PR Edit** tab. The PR Edit page will be displayed with fields populated from the existing PR name, but having the new PR name chosen above.
 - See Section 14.1.2 on using the PR Edit page and saving any changes made

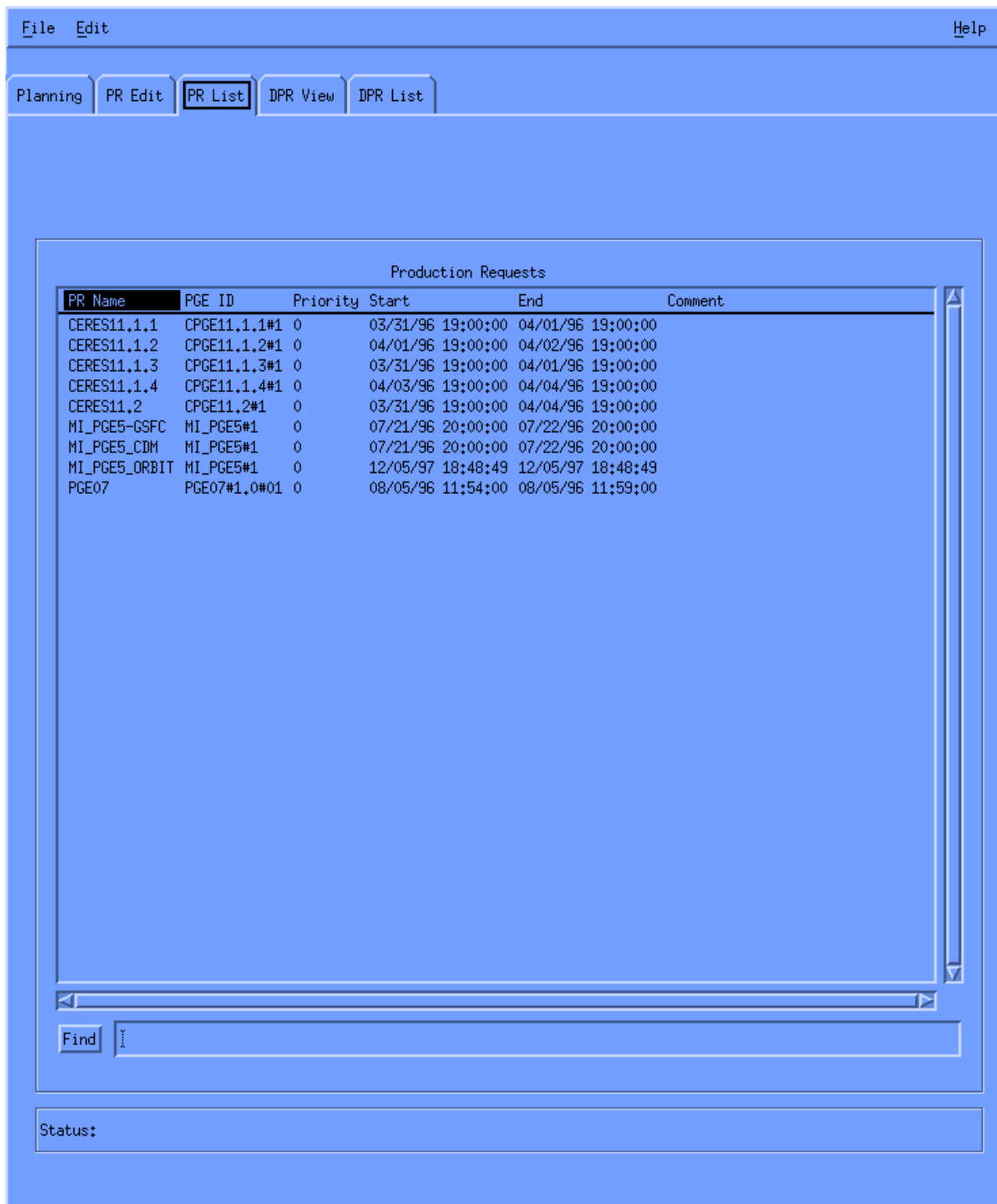


Figure 14.1.3-1. PR List GUI.

Table 14.1.3-1. Viewing Production Requests - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	PR List	click tab
2	(Optional) <i>search string</i>	click Find button
3	(Optional) <i>PGE</i>	click PGE

14.1.4 Viewing Data Processing Requests

Clicking on the DPR View GUI tab displays a list of all DPRs for all PRs entered into the system.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

To invoke the DPR View GUI, execute the procedure steps that follow:

- 1 From the PR Editor GUI, click on **DPR View** tab. The following information will be displayed:
 - Data Processing Request Identification.
 - PGE ID and its parameters.
 - Request Data and Status.

Table 14.1.4-1. Viewing Data Processing Requests - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	DPR View tab	Click on tab

14.1.5 Listing Data Processing Requests

Selection of one PR on the PR List by highlighting it and then clicking on the DPR List tab, brings up a detailed display of all DPRs associated with the selected PR. These may be examined in order to develop production plans and schedule jobs.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

To invoke the DPR List GUI, execute the procedure steps that follow:

- 1 From the PR Editor GUI, click on **DPR List** tab. The following information is displayed:
 - DPR Id.
 - PGE Id.
 - PR Name.
 - Data Start Time.

- Data Stop Time.

Table 14.1.5-1. Listing Data Processing Requests - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	DPR List tab	Click on tab

14.2 Using the Production Planning Workbench

The Production Planner uses the Production Planning Workbench to create new production plans and display a planning timeline.

14.2.1 Using the Planning Workbench to Run One PGE

Once a PGE has been fully registered (Section 13.1), its test data files have been inserted to the Science Data Server (Section 13.2), and a single Data Processing Request (DPR) has been generated, the Planning Workbench can be used to plan for one execution run of a single PGE.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The required UNIX environment variables have been set properly.
2. The required servers of the ECS System are up and running.
3. A DPR has been generated successfully.

To use the Planning Workbench to run one PGE, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then telnet to a PLN Host.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the PLN Host.
 - It is recommended that this procedure begin within a new command shell on a PLN Host.
 - Change the working directory to the location where the script to run the Planning Workbench is located. (e.g. /usr/ecs/TS1/CUSTOM/utilities).
- 2 At the UNIX prompt on an PLN Host, Type **EcPIStartAll mode application_id**, press **Return**.
 - The **mode** is one of modes used in the ECS system. e.g. TS1.
 - The **application_id** is a numerical number. e.g. 1.
 - For example: **EcPIStartAll TS1 1, Return**.
 - A Planning Workbench GUI will be appeared as shown in **Fig. 14.2.1-1**.

- 3 In the Planning Workbench GUI, go to the subwindow labeled **Unscheduled** and click on a Production Request name.
 - The Production Request name is the name under which the PR was saved in Section 14.1.2.
 - The PR name entry will be highlighted.
- 4 In the Planning Workbench GUI, click on the button next to the label **Schedule** (the button has an inverted triangle on it).
 - The PR highlighted in step 3 will appear in the subwindow labeled **Scheduled**.
- 5 In the Planning Workbench GUI, click on the **Activate** button
 - A small GUI labeled **Plan Activation** will be displayed.
- 6 In the Plan Activation GUI, set the time in the time field forward to allow ample time for the PGE to run. Then click on the **Ok** button.
 - All that is necessary is for there to be sufficient time for the PGE run. There is no penalty for allowing *too* much time.
 - The Production Request thus planned will be submitted to processing and its progress can be monitored with AutoSys (see Section 14.3).
- 7 When tasks are completed with the Planning Workbench GUI, click on the **File** menu, then choose **Exit**.
 - The Planning Workbench GUI will disappear.

Table 14.2.1-1. Using the Planning Workbench to Run One PGE - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Tools → <u>X</u> term	telnet to PLN Host
2		press Return
3	<i>PR name</i>	click PR name
4	Schedule	click button
5	Activate	click button
6	<i>time</i>	adjust time and click Ok
7	<u>F</u> ile → <u>E</u> xit	(No action)

File
Options
Help

Plan Name: NONAME
Status: CANDIDATE
Strategy Id:

Baseline
Activate
Kill

Rollover Time:

Comments:

Production Requests

Unscheduled:

NAME	PRIORITY
CERES11.1.1	0
CERES11.1.2	0
CERES11.1.3	0
CERES11.1.4	0
MI_PGE5-GSFC	0
MI_PGE5_CDM	0
MI_PGE5_ORBIT	0

schedule:
unschedule:

Scheduled:

NAME	PRIORITY
------	----------

Prioritize
Refresh

Figure 14.2.1-1. Planning Workbench GUI.

14.3 Monitoring Production

The progress of one or more PGEs running within the PDPS may be monitored. The COTS tool used for this purpose is AutoSys® by Atria Software. Each Data Processing Request results in seven AutoSys jobs that are boxed together. An AutoSys job name follows the template:

PGName#versionMMDDYYhhmm.Suffix

where *PGName* is replaced by the name of the PGE; *version* is replaced by the version of the PGE; *MMDDYY* is replaced by the two-digit month (*MM*), day (*DD*), and year (*YY*) that the job is scheduled to run; *hhmm* is replaced by the time (hours and minutes) that the job is scheduled to run; and *Suffix* is a two character extension indicating the job phase of the DPR. Refer to Table 14.3-1.

For example, for version 2 of a PGE named MOPITT4 scheduled to run on December 8, 1998 at 1:00pm, the AutoSys jobs making up that DPR would be:

MOPITT4#20818981300.Al
MOPITT4#20818981300.St
MOPITT4#20818981300.Pr
MOPITT4#20818981300.EX
MOPITT4#20818981300.Ps
MOPITT4#20818981300.Ds
MOPITT4#20818981300.Da

Table 14.3-1. AutoSys Jobs for a DPR

Job Name Suffix	Description
.Al	Resource allocation
.St	Staging
.Pr	Pre-processing
.EX	Execution of the PGE itself
.Ps	Post-processing
.Ds	De-staging
.Da	De-allocation of resources

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The required UNIX environment variables have been set properly. In particular, the environment variables AUTOSYS, AUTOUSER, and AUTOSERV must be set.

To monitor production, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then telnet to a PLN Host.

- Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the PLN Host.
 - It is recommended that this procedure begin within a new command shell on a PLN Host.
 - Change the working directory to the location where the script for starting the Autosys is stored. (e.g. /data1/autotreeb/autouser).
 - Source the environmental file. (**source** autosys.env.taltos).
- 2 At the UNIX prompt on the PLN Host, type **autosc &**, press **Return**.
- A GUI labeled **AutoSys** will be displayed.
 - This GUI will contain eight buttons for invoking various tools available under AutoSys.
- 3 In the AutoSys GUI, click on the **Ops Console** button.
- A GUI labeled **AutoSys Job Activity Console** GUI will be displayed.
 - The main subwindow of this GUI will contain a dynamically updated list of AutoSys jobs (seven jobs make up one DPR) currently scheduled.
 - To disable dynamic updating of the main subwindow (which may be distracting), click on **Freeze Frame** in the small subwindow labeled **Show**.
 - DPRs will be listed in the column labeled **Job Name** and their statuses (SUCCESS, FAILURE, TERMINATED) will be listed in the column labeled **Status**.
 - To view job status information for a particular DPR, click on a DPR. Below the main subwindow, available job status information will be displayed.
 - To view the existing event report, under the label **Reports**, click on the middle diamond labeled **Event**. The current event status for the selected DPR will be displayed in the subwindow labeled **Event Report**.
 - Exit the **AutoSys Job Activity Console** by clicking on the **Exit** button.
- 4 In the AutoSys GUI, click on the **TimeScape** button.
- A GUI labeled **TimeScape** GUI will be displayed.
 - The main subwindow labeled **Job Name** of this GUI will contain a dynamically updated list of AutoSys jobs (seven per DPR) currently scheduled.
 - To disable dynamic updating of AutoSys jobs (which may be distracting), click on the **Freeze Frame** button.
 - The color of each job indicates its status according to the legend on the left side of the GUI.
 - The time line is shown on the right side of the GUI with time marked at the top. A red vertical line (dashed) indicates the current time.
 - Exit the **TimeScape** GUI by clicking on the **File** menu and selecting **Exit**.
- 5 To quit AutoSys, in the AutoSys GUI, click on the **Exit** button.
- The AutoSys GUI will disappear.

Table 14.3-2. Monitoring Production - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>X</u> term	telnet to PLN Host
2	autosc &	press Return
3	Ops Console	click Ops Console button
4	TimeScape	click TimeScape button
5	Exit	click Exit button

14.4 Using the Q/A Monitor

The Q/A Monitor allows the output products produced during a PGE run to be accessed and examined. Input test data granules and Production History files can be retrieved in the same manner. The Q/A Monitor retrieves output products based on the collection name (*i.e.* the ESDT) and time of Insertion to the Science Data Server.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The required UNIX environment variables have been set properly.
2. The desired output products have been successfully Inserted to the Science Data Server.

To use the Q/A Monitor, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then telnet to a PLN Host.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the PLN Host.
 - It is recommended that this procedure begin within a new command shell on a PLN Host.
 - Change the working directory to the location where the script for starting the Q/A Monitor GUI is stored.
- 2 At the UNIX prompt on the PLN Host, type ***EcDpPrQaMonitorGUI mode application_id***, press **Return**.
 - The ***mode*** is the operations mode.
 - The Q/A Monitor GUI will be displayed.
 - For example, type ***EcDpPrQaMonitorGUI TS1 1***; press **Return**.
 - Various messages from the Q/A Monitor will appear in this window as it is running.
- 3 In the Q/A Monitor subwindow labeled **Data Types**, select an ESDT from the list presented and click on it.
 - Use the scroll bars if necessary to locate desired ESDT.

- Optionally, use the **Find** field and button to locate an ESDT.
- 4 In the Q/A Monitor, under the label **Data Granule Insert Date (mm/dd/yy)**, set the date range within which the search for granules of the ESDT selected will be conducted.
 - The date range can be made arbitrarily large to select all granules of a particular collection (ESDT).
 - The dates refer to date of granule Insert to the Science Data Server.
 - 5 In the Q/A Monitor, click on the **Query** button.
 - The results of the query will be displayed in the bottom window, labeled **Data Granules**.
 - All granules having the ESDT selected in step 3 and having Insert times within the date range specified in step 4 will be listed in this window.
 - 6 In the Q/A Monitor subwindow labeled **Data Granules**, click on one of the data granules listed to be examined.
 - The data granule selected will be highlighted.
 - 7 To retrieve the data granule's Production History file, click on the **Retrieve Prod History** button.
 - The Production History (PH) tar file corresponding to the selected data granule will be retrieved from the Science Data Server and placed on the local machine (a PLN Host) in the directory /var/tmp.
 - The PH can then be moved or copied manually from the /var/tmp directory to a user working directory for examination (see Section 16.1.2).
 - Only the PH file is retrieved with the **Retrieve Prod History** button.
 - If only the PH is desired, exit this procedure. To retrieve the data granule itself, continue on to step 8.
 - 8 To retrieve the data granule, click on the **Retrieve Data Granule** button and note its file name (listed in the entry for the granule; you may have to scroll over to the right to see it).
 - The data granule selected will be retrieved from the Science Data Server and placed on the local machine (a PLN Host) in the directory /var/tmp.
 - A granule of any format (binary, ASCII, HDF, HDF-EOS) may be retrieved in this manner. Only HDF and HDF-EOS granules, however, may be further visualized using EOSView as described in the next steps.
 - 9 To examine the data granule, click on the **Visualize data** tab.
 - The **Visualize data** page will be displayed.
 - 10 In the main subwindow on the **Visualize data** page, locate the file name of the granule retrieved in step 8 and click on it.
 - The item selected will be highlighted and will appear in the **Selection** subwindow below.

- 11 Click on the **Visualize** button.
 - This action will invoke EOSView with the granule selected.
 - The granule must be HDF or HDF-EOS format.
 - See Sections 16.1 and 16.1 for use of EOSView.
- 12 When tasks are completed with the Q/A Monitor GUI, click on the **File** menu, then choose **Exit**.
 - The Q/A Monitor GUI will disappear.

Table 14.4-1. Using the Q/A Monitor - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>X</u> term	telnet to PLN Host
2	<i>EcDpPrQaMonitorGUI mode application_id</i>	press Return
3	<i>data type</i>	click entry
4	<i>data granule insert date</i>	adjust dates
5	Query	click Query button
6	<i>data granules</i>	click entry
7	(Optional) Retrieve Prod History	click button
8	Retrieve Data Granule	click button
9	Visualize data	click tab
10	<i>granule file name</i>	click entry
11	Visualize	click button
12	<u>F</u> ile → <u>E</u> xit	(No action)

15. File Comparison

An important part of SSI&T is verifying that the output files produced at the DAAC are identical (within particular tolerances) to the test output files delivered with the DAPs. A successful comparison is a strong indication that the porting of the science software from the development facility at the SCF to the operational facility at the DAAC has not introduced any errors.

A number of file comparison tools are available during SSI&T via the SSIT Manager GUI or they can be invoked from the UNIX command line. Two tools are available for comparing HDF or HDF-EOS files, one tool for comparing ASCII files, and another tool for assisting in comparing binary files.

It is assumed that the Instrument Team has delivered test output files (produced at their SCF) with which to perform the comparison.

15.1 Using the GUI HDF File Comparison Tool

Most output files (products) from PGEs run in the DAAC will be in HDF-EOS format. The GUI File Comparison Tool allows comparison of two HDF or HDF-EOS files.

Table 15.1-1. Assumptions and Limitations

Order	Assumption/Limitation
1	Two HDF or HDF-EOS files exist with similar structures, or at least having a data set in common.
2	The SSIT Manager is running.
3	If either of the two HDF/HDF-EOS files is in a ClearCase VOB, a ClearCase view was set before the SSIT Manager was started.

To compare two HDF or HDF-EOS files, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **P**roduct Examination, then **H**DF.
 - The HDF File Comparison Tool GUI will be displayed.
- 2 In the HDF File Comparison Tool GUI, click on the **F**ile 1 button.
 - Read the *Systems Description* document and the *Operations Manual*. Both of these or their equivalent should be in the delivery.

Table 15.1-2. Using the GUI HDF File Comparison Tool - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	(No entry)	read all documentation
2	<u>T</u> ools → <u>X</u> term	telnet to SPR SGI

15.2 Using the hdiff HDF File Comparison Tool

Most output files (products) from PGEs run in the DAAC will be in HDF-EOS format. The *hdiff* File Comparison Tool is a text-oriented tool run from the command line. It allows comparison of two HDF or HDF-EOS files.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Table 15.2-1. Assumptions and Limitations

Order	Assumption/Limitation
1	Two HDF or HDF-EOS files exist with similar structures, or at least having a data set in common.
2	The SSIT Manager is running.
3	If either of the two HDF/HDF-EOS files is in a ClearCase VOB, a ClearCase view was set before the SSIT Manager was started.

To compare two HDF or HDF-EOS files, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **P**roduct Examination, then **F**ile Comparison, and then **H**DF (**hdiff**).
 - An xterm window running *hdiff* will be displayed.
- 2 In the xterm window at the prompt **Options? (-h for help)**, type in any desired options and then press **Return**.
 - To see the list of available options, type **-h** and press **Return** to the prompt.
- 3 In xterm window at the prompt **1st file to compare?**, type *filename1*, press **Return**.
 - The *filename1* is the file name of the first of two HDF or HDF-EOS files to be compared.
 - If *filename1* is not in the current directory (the directory from which the SSIT Manager was run), include the full path name with the file name.
- 4 In xterm window at the prompt **2nd file to compare?**, type *filename2*, press **Return**.
 - The *filename2* is the file name of the second of two HDF or HDF-EOS files to be compared.
 - If *filename2* is not in the current directory (the directory from which the SSIT Manager was run), include the full path name with the file name.

- The two files will be compared and the output will be displayed in the xterm window.

Table 15.2-2. Using the hdiff HDF File Comparison Tool - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>P</u> roduct Examination → <u>F</u> ile Comparison → <u>H</u> DF (hdiff)	(No action)
2	<i>filename1</i>	press Return
3	<i>filename2</i>	press Return
4	(No entry)	visually compare the two files

15.3 Using the ASCII File Comparison Tool

Most output files (products) from PGEs run in the DAAC will be in HDF-EOS format. A small minority may be in ASCII (text) format. The ASCII File Comparison Tool is a front-end to *xdiff* UNIX X Window tool for comparing two ASCII files.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Table 15.3-1. Assumptions and Limitations

Order	Assumption/Limitation
1	Two ASCII files exist and have read permissions.
2	If either of the two ASCII files is in a ClearCase VOB, a ClearCase view was set before the SSIT Manager was started.

To compare two ASCII files, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **Tools** menu, then choose **Product Examination**, then **File Comparison**, and then **ASCII**.
 - An xterm window running *xdiff* will be displayed.
- 2 In xterm window at the prompt **1st file to compare?**, type *filename1*, press **Return**.
 - The *filename1* is the file name of the first of two ASCII files to be compared.
 - If *filename1* is not in the current directory (the directory from which the SSIT Manager was run), include the full path name with the file name.
- 3 In xterm window at the prompt **2nd file to compare?**, type *filename2*, press **Return**.
 - The *filename2* is the file name of the second of two ASCII files to be compared.
 - If *filename2* is not in the current directory (the directory from which the SSIT Manager was run), include the full path name with the file name.
 - A GUI labeled **xdiff** will be displayed.

- 4 In the GUI labeled **xdiff**, view the differences between the two files displayed.
 - File *filename1* will be displayed on the left side of the GUI. File *filename2* will be displayed on the right.
 - Only sections of file in which there are differences will be displayed. A “bang” character (!) at the beginning of a line indicates that a difference was found.
 - For further help on *xdiff*, type **man xdiff**, press **Return** in an xterm window.
 - Close the display window by using the pull down menu from the X window in the upper left corner.
- 5 In the xterm window at the prompt **Hit return for another diff, 'q <returns> to quit:**, type **q** and then press **Return** to quit or just press **Return** to perform another comparison.

Table 15.3-2. Using the ASCII File Comparison Tool - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>P</u> roduct Examination → <u>F</u> ile Comparison → <u>A</u> SCII	(No action)
2	<i>filename1</i>	press Return
3	<i>filename2</i>	press Return
4	(No entry)	visually compare the two files
5	q Return	press Return

15.4 Using the Binary File Difference Assistant

Most output files (products) from PGEs run in the DAAC will be in HDF-EOS format. A small minority may be in some binary format. The Binary File Difference Assistant aids the user in constructing code that allows comparison of binary output files. Since there is an unwieldy number of possibilities for binary file formats, this tool cannot compare two binary files without some custom code written at the DAAC, hence, the “Assistant” in the name. The Binary File Difference Assistant aids the user by generating a makefile, a driver module, and a template comparison module in C, FORTRAN 77 or IDL (Interactive Data Language). The user then edits these templates to read the particular binary format in question according to a SCF-supplied format specification.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Table 15.4-1. Assumptions and Limitations

Order	Assumption/Limitation
1	Two binary files exist with read permissions.
2	Information on the data structure and tolerances have been provided by the Instrument Team.

To compare two binary files, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **P**roduct Examination, then **F**ile Comparison, and then **B**inary.
 - The Binary File Difference Assistant tool GUI will be displayed.
- 2 In the Binary File Difference Assistant tool GUI, click on one of the languages listed under the **Select Language** label. The choices are C, FORTRAN, or IDL.
 - The choice of language depends largely on preference. It does not necessarily have to be the language that was used to create the files being compared.
- 3 Optionally, click on either the **Image** button or the **Structure** button located under the label **Compare Function**.
 - Clicking on the **Image** button will display a code example for comparing binary files containing images.
 - Clicking on the **Structure** button will display a code example for comparing binary files containing structures or records.
 - The displayed listing well documented and should be read.
 - The language of the code will depend on the language selection made in step 2.
- 4 Optionally, click on either the **Image** button or the **Structure** button located under the label **Driver**.
 - Clicking on the **Image** button will display a code example for a driver invoking the compare function for binary files containing images.
 - Clicking on the **Structure** button will display a code example for a driver invoking the compare function for binary files containing structures or records.
 - The displayed listing well documented and should be read.
 - The language of the code will depend on the language selection made in step 2.
- 5 Optionally, click on either the **Help** button.
 - A Help GUI will be displayed.
 - To end help, click on the **Dismiss** button.
 - The Help GUI may remain displayed while using the Binary File Difference Assistant.
- 6 Once familiar with the code examples (steps 3 and 4), click on the **Copy** button.
 - A GUI labeled **Enter Unique ID** will be displayed.
 - In the field labeled **Enter unique file identifier:**, type *fileID*, click on the **OK** button.
 - The *fileID* will be used in the file names of the files copied over. These files will be:
 - **C:**
 - DaacBinDiff_*fileID*.c Compare function
 - DaacBinDiff_*fileID*_driver.c Driver
 - DaacBinDiff_*fileID*.mak Makefile
 - **FORTTRAN:**
 - DaacBinDiff_*fileID*.f Compare function
 - DaacBinDiff_*fileID*_driver.f Driver
 - DaacBinDiff_*fileID*.mak Makefile
 - **IDL:**

- DaacBinDiff_*fileID*.pro Compare function
 - DaacBinDiff_*fileID*_driver.pro Driver
 - DaacBinDiff_*fileID*.sh Shell script with here document
- The files will be copied into the directory from which the SSIT Manager is being run.
- 7 Using any desired text editor, customize the files for the job at hand. Then build the executable using the customized makefile provided (for C and FORTRAN). Then run the program to perform the binary file comparison.

Table 15.4-2. Using the Binary File Difference Assistant - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>P</u> roduct Examination → <u>F</u> ile Comparison → <u>B</u> inary	(No action)
2	language	click
3	(Optional) Compare Function → Image Structure	click
4	(Optional) Driver → Image Structure	click
5	(Optional) Help	click
6	(No entry)	customize files

16. Data Visualization

In order to fully ascertain the success of science software in producing scientifically valid data sets, the data need to be displayed in forms that convey the most information easily. Data visualization enables this to be done. This section describes the data visualization tools available at the DAACs during SSI&T and how to use them for detailed inspection of data sets produced by PGEs. Only some aspects of data visualization will be addressed in these procedures. For further information, see the references below.

The two visualization tools provided at the DAACs are EOSView and IDL. EOSView was developed by ECS and is a user-friendly GUI for creating two-dimensional displays from HDF-EOS objects (Grid, Swath) as well as the standard HDF objects (SDS, Vdata, Image, Text). It has additional features such as thumbnail-panning, colorization, zooming, plotting, animation, and flatfile output.

IDL (Interactive Data Language) is a COTS display and analysis tool widely applied in the scientific community. It is used to create two-dimensional, three-dimensional (volumetric), and surface/terrain displays from binary, ASCII, and many other CSDTs (or formats) in addition to HDF. IDL has additional features including flexible input/output, custom colorization, plotting, scripting, raster-math, geographic registration, map projection transformation, and vector map overlay.

16.1 Viewing Product Metadata with the EOSView Tool

This procedure describes how to use the EOSView tool to inspect the metadata in the HDF-EOS output file from a PGE. See Section 14.2 for how to inspect the science data. An HDF-EOS file is defined a file produced using the SDP Toolkit metadata tools and having, at a minimum, the ECS core metadata.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools.
2. EOSView has been properly installed and is accessible to the user.

To view product metadata with the EOSView tool, execute the procedure steps that follow:

- 1 a From the SSIT Manager, click on the **T**ools menu, then choose **P**roduct Examination, then **E**OSView.
 - The EOSView GUI will be displayed.

- 1 b from Alternately, if EOSView isn't available from the SSIT Manager GUI, invoke EOSView from the command-line.
 - Go to the proper area by typing `cd /usr/ecs/TS1/CUSTOM/eosview` <RETURN>
 - Start EOSView by typing `EOSView` <RETURN>
- 2 In the GUI labeled **EOSView - EOSView Main Window**, click on the **File** menu and select **Open**.
 - The **Filter** GUI will be displayed.
- 3 In the subwindow labeled **Filter**, enter full path name and file name wildcard template. For example, enter `/home/MyDirectory/MySubdirectory/*`.
 - The `/home/MyDirectory/MySubdirectory/*` represents the location to the directory containing the HDF-EOS files to examine.
 - The asterisk (*) is a wildcard template that represents all files in that directory; other wildcard templates can narrow the search further, *e.g.* `*.hdf`.
 - Use the **Directories** field to further select the correct directory.
 - Files found matching the wildcard template in the chosen directory will be displayed in **Files** subwindow.
- 4 In the **Files** subwindow, click on the file name of the HDF-EOS file to examine. Then click on the **OK** button.
 - A GUI labeled **EOSView - MyOutputFile.hdf** will be displayed where *MyOutputFile.hdf* is the file name of the file chosen in step 3.
 - Be patient - this GUI may take some time to appear, particularly for large files.
 - Once displayed, a list of HDF objects will appear in the main window. If nothing is listed, it means that no HDF objects were found within the file.
- 5 In the GUI labeled **EOSView - MyOutputFile.hdf**, click on an object listed for which metadata is to be inspected.
 - The object selected will be highlighted.
 - Do not double click on object since this will cause a **Dimension** GUI to be displayed instead.
- 6 In the GUI labeled **EOSView - MyOutputFile.hdf**, click on the **Attributes** menu and select **Global**.
 - A GUI labeled **EOSView - Text Display** will be displayed.
 - The global metadata associated with the object selected (in step 5) will be displayed in a scrollable field.
 - If instead, the message "Contains no Global Attributes" appears, then the selected object contains no global metadata.
- 7 Repeat steps 5 and 6 for each HDF object within the selected HDF-EOS file for which metadata is to be examined.
- 8 In the GUI labeled **EOSView - MyOutputFile.hdf**, click on the **File** menu and select **Close**.
 - The **EOSView - MyOutputFile.hdf** GUI will disappear.
 - Be patient - this GUI may take some time to disappear, particularly for large files.

- 9 In the GUI labeled **EOSView - EOSView Main Window**, click on the **File** menu and select **Exit**.
 - The **EOSView - EOSView Main Window** GUI will disappear.

Table 16.1-1. Viewing Product Metadata with the EOSView Tool - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1a	<u>T</u> ools → <u>P</u> roduct Examination → <u>E</u> OSView	(No action)
1b	Alternately, from command line	
	cd /usr/ecs/TS1/CUSTOM/eosview	press Return
	EOSView	press Return
2	File → Open	(No action)
3	/home/MyDirectory/MySubdirectory/*	select file filter
4	OK	select file
5	select object	click
6	Attributes → Global	(No action)
7	(No entry)	repeat 5 and 6
8	File → Close	(No action)
9	File → Exit	(No action)

16.2 Viewing Product Data with the EOSView Tool

This procedure describes how to use the EOSView tool to inspect the science data in the HDF-EOS output file from a PGE. See Section 12.1 for how to inspect the metadata. An HDF-EOS file is defined as a file produced using the SDP Toolkit metadata tools and having, at a minimum, the ECS core metadata.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools.
2. EOSView has been properly installed and is accessible to the user.

To view product data with the EOSView tool, execute the procedure steps that follow:

- 1 From the SSIT Manager, click on the **Tools** menu, then choose **Product Examination**, then **EOSView**.
 - The EOSView GUI will be displayed.
- 2 In the GUI labeled **EOSView - EOSView Main Window**, click on the **File** menu and select **Open**.
 - The **Filter** GUI will be displayed.

- 3 In the **Filter** subwindow, enter full path name and file name wildcard template. For example, enter */home/MyDirectory/MySubdirectory/**.
 - The */home/MyDirectory/MySubdirectory/** represents the location to the directory containing the HDF-EOS files to examine.
 - The asterisk (*) is a wildcard template that represents all files in that directory; other wildcard templates can narrow the search further, *e.g.* **.hdf*.
 - Use the **Directories** field to further select the correct directory.
 - Files found matching the wildcard template in the chosen directory will be displayed in **Files** subwindow.
- 4 In the **Files** subwindow, click on the file name of the HDF-EOS file to examine. Then click on the **OK** button.
 - A GUI labeled **EOSView - MyOutputFile.hdf** will be displayed where *MyOutputFile.hdf* is the file name of the file chosen in step 3.
 - Be patient - this GUI may take some time to appear, particularly for large files.
 - Once displayed, a list of HDF objects will appear in the main window. If nothing is listed, it means that no HDF objects were found within the file.
- 5 In the GUI labeled **EOSView - MyOutputFile.hdf**, double click on an object listed for which data is to be inspected.
- 6 Go to procedure depending upon the type of the object selected in step 5.
 - Proceed to Section 16.2.1 if object is an HDF Image.
 - Proceed to Section 16.2.2 if object is an HDF-EOS Grid.
 - Proceed to Section 16.2.3 if object is an HDF-EOS Swath.
 - Proceed to Section 16.2.4 if object is an HDF SDS.
 - Proceed to Section 16.2.5 if object is in another format.
- 7 Repeat steps 5 and 6 for each HDF object within the selected HDF-EOS file for which data is to be examined.
- 8 In the GUI labeled **EOSView - MyOutputFile.hdf**, click on the **File** menu and select **Close**.
 - The **EOSView - MyOutputFile.hdf** GUI will disappear.
 - Be patient - this GUI may take some time to disappear, particularly for large files.
- 9 In the GUI labeled **EOSView - EOSView Main Window**, click on the **File** menu and select **Exit**.
 - The **EOSView - EOSView Main Window** GUI will disappear.

**Table 16.2-1. Viewing Product Data with the EOSView Tool -
Quick-Step Procedures**

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>P</u> roduct Examination → <u>E</u> OSView	(No action)
2	File → Open	(No action)
3	/home/MyDirectory/MySubdirectory/*	select file filter
4	OK	select file
5	select object	double click
6	(No entry)	go to procedure 14.2.1, 14.2.2, 14.2.3, 14.2.4, or 14.2.5
7	(No entry)	repeat 5 and 6
8	File → Close	(No action)
9	File → Exit	(No action)

16.2.1 Viewing HDF Image Objects

This procedure describes how to use the EOSView tool to view science Images (typically, browse images) in the HDF-EOS output file from a PGE. See Section 14.2 for how to select an HDF Image object within an HDF-EOS file. An HDF-EOS file is defined as a file produced using the SDP Toolkit metadata tools and having, at a minimum, the ECS core metadata.

Once an HDF Image is displayed, a number of options are available. These include colorization, zooming, panning, and animation. Each is described in the procedures below as an optional step.

EOSView will display and HDF Image in its referenced default color palette if the file contains one. If the display looks “blackened out”, it may mean that no default color palette is available or referenced by the Image object. In this case, a palette will have to be selected by the user.

Zooming allows both zoom in and zoom out according to a Bilinear Interpolation or Nearest Neighbor resampling method. Bilinear Interpolation involves averaging to produce a “smoothed” display appropriate for gray scale band Images or derived geophysical parameter Images (*e.g.* temperature, vegetation index, albedo). Nearest Neighbor produces a non-smoothed display appropriate for derived thematic image-maps (*e.g.* land cover, masks).

Panning allows user to select the portion of a zoomed Image to be displayed.

Animation allows multiple Images to be displayed in an animated fashion in a number of modes. Stop-at-End mode allows Images to be displayed to the end just once. Continuous Run mode lets the animation run through the Images repeatedly and Bounce mode does a forward/reverse run through any set of Images repeatedly. All animation runs can be stopped at any time and then resumed in either the forward or reverse directions. The speed of an animation can be adjusted as well.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools and that at least one object is an HDF image (RIS8, RIS24, *i.e.* Browse data).
2. EOSView has been properly installed and is accessible to the user.
3. The HDF-EOS file has been read into EOSView by the procedures described in Section 14.2.

To view an HDF-EOS Image object with the EOSView tool, execute the procedure steps that follow:

- 1 In the GUI labeled **EOSView - MyOutputFile.hdf**, double click on an Image object listed for which data is to be inspected.
 - A GUI labeled **EOSView - Image Display Window - MyImageObject** will be displayed where *MyImageObject* will be replaced by the name of the object selected as listed. For example, **EOSView - Image Display Window - Image [512x512] ref=2 (palette)**.
 - Be patient - this GUI may take some time to appear, particularly for large files.
- 2 Optional colorization. In the GUI labeled **EOSView - Image Display Window - MyImageObject**, click on the **Palette** menu, then select **Select** and then select one of the palettes listed: **Default**, **Greyscale**, **Antarctica**, **Rainbow**, or **World Colors**.
 - The selection of palette will not result in any change to the data in the file or to the object's default palette.
 - This selection may be repeated until the desired palette is chosen.
- 3 Optional zooming. In the GUI labeled **EOSView - Image Display Window - MyImageObject**, click on the **Zooming** menu, then select **Select** and then select one of the resampling methods listed: **Bilinear Interpolation** or **Nearest Neighbor**. Then click on the **Zoom In** or **Zoom Out** buttons to apply the method.
 - The selection of resampling method and zoom will not result in any change to the data in the file.
 - The zooming options may be repeated as desired.
- 4 Optional panning while zooming. In the GUI labeled **EOSView - Image Display Window - MyImageObject**, a thumbnail representation of the entire Image will be displayed in the subwindow labeled **Pan Window**. A hollow rectangle on the thumbnail indicates the portion of the Image that is being displayed in the main window. Use the mouse left button to click and drag the rectangle to a new location on the thumbnail image.
 - The portion of the zoomed Image shown in the main window will be the portion indicated by the hollow rectangle on the thumbnail image.
 - The panning will not result in any change to the data in the file.
 - The panning option may be repeated as desired.
- 5 To end the session with colorization, zooming, or panning, in the GUI labeled **EOSView - Image Display Window - MyImageObject**, click on the **File** menu and select **Close**.

- The **EOSView - Image Display Window - MyImageObject** GUI will disappear.
- 6 Optional animation. In the GUI labeled **EOSView - MyOutputFile.hdf**, click on the **Options** menu, then select **Animated images**.
- A GUI labeled **EOSView - Image Animation Window - MyOutputFile.hdf** will be displayed.
 - Be patient - this GUI may take some time to appear, particularly for large files.
 - Optionally, click on the **Palette** menu to select a palette.
 - Optionally, click on the **Options** menu and then select **Mode** to select how the animation is to be run. Choose **Stop at end**, **Continuous run**, or **Bounce**.
 - Click on the Stop button, denoted by the || symbol (center) to halt the animation.
 - Resume the animation by clicking on either the Forward Play (denoted by the >> symbol) or the Reverse Play (denoted by the << symbol).
 - There is also Forward Increment (>|) and Reverse Increment (|<) button.
 - The animation speed may be adjusted by moving the **Speed** slider in either the “+” or “-” direction.
 - To end animation session, click on the **File** menu and then select **Close**.

Table 16.2.1-1. Viewing HDF Image Objects - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	select Image object	double click
2	(Optional) P <u>a</u> lette → Select → Default Greyscale Antarctica Rainbow World Colors	choose palette
3	(Optional) Z <u>o</u> oming → Select → Bilinear Interpolation Nearest Neighbor	choose resampling mode
4	(Optional) Pan Window	click and drag
5	File → Close	(No action)
6	(Optional) O <u>p</u> tions → Animated images	adjust as desired

16.2.2 Viewing HDF-EOS Grid Objects

This procedure describes how to use the EOSView tool to view science data in the HDF-EOS output file that are in HDF-EOS Grid format. These are generally the science data and not browse images). See Section 14.2 for how to select an HDF-EOS Grid object within an HDF-EOS file. An HDF-EOS file is defined as a file produced using the SDP Toolkit metadata tools and having, at a minimum, the ECS core metadata.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools and that at least one object is an HDF-EOS Grid.
2. EOSView has been properly installed and is accessible to the user.
3. The HDF-EOS file has been read into EOSView by the procedures described in Section 14.2.

To view an HDF-EOS Grid object with the EOSView tool, execute the procedure steps that follow:

- 1 In the GUI labeled **EOSView - MyOutputFile.hdf**, double click on an Grid object listed for which data is to be inspected.
 - A GUI labeled **EOSView - Grid Select** will be displayed.
 - Be patient - this GUI may take some time to appear, particularly for large Grid objects.
 - Clicking on the **Grid Information** checkbox and then clicking on the **OK** button displays grid information about the selected Grid object such as X-Dimension value, Y-Dimension value, Upper Left Point values, and Lower Right Point values.
 - Clicking on the **Projection Information** checkbox and then clicking on the **OK** button displays projection information about the selected Grid object such as the Projection Name and the Zone Code.
 - Clicking on the **Dimensions** checkbox and then clicking on the **OK** button displays dimension information about the selected Grid object such as the Dimension Names and Sizes.
 - Clicking on the **Attributes** checkbox and then clicking on the **OK** button displays attribute information about the selected Grid object such as the attribute names and values.
- 2 In the GUI labeled **EOSView - Grid Select**, click on the **Data Fields** checkbox and then click on the **OK** button. Then double click on one of the data fields listed.
 - A GUI labeled **EOSView - Grid - GridObjectName - Start/Stride/Edge** will be displayed where *GridObjectName* will be replaced by the name of the Grid object selected in step 1.
- 3 In the GUI labeled **EOSView - Grid - GridObjectName - Start/Stride/Edge**, click on the checkboxes for both **YDim** and **XDim** and then click on the **OK** button.
 - A GUI labeled **MyDataField** will be displayed where *MyDataField* will be replaced by the name of the data field selected in step 2.
 - The data will be displayed in this GUI in the form of a table of values.
- 4 In the GUI labeled **MyDataField**, click on the **File** menu and then select **Make Image**. Optionally adjust the default minimum and maximum data values and then click on the **Continue** button.
 - Adjusting the minimum and maximum data values will affect only the display (contrast) and not the actual data in the file.

- A GUI labeled **EOSView - Swath/Grid Image** will appear, displaying the data field in image form.
- 5 Optional colorization. In the GUI labeled **EOSView - Swath/Grid Image**, click on the **Palette** menu, then select **Select** and then select one of the palettes listed: **Default**, **Greyscale**, **Antarctica**, **Rainbow**, or **World Colors**.
 - The selection of palette will not result in any change to the data in the file or to the object's default palette.
 - This selection may be repeated until the desired palette is chosen.
 - 6 Optional zooming. In the GUI labeled **EOSView - Swath/Grid Image**, click on the **Zooming** menu, then select **Select** and then select one of the resampling methods listed: **Bilinear Interpolation** or **Nearest Neighbor**. Then click on the **Zoom In** or **Zoom Out** buttons to apply the method.
 - The selection of resampling method and zoom will not result in any change to the data in the file.
 - The zooming options may be repeated as desired.
 - 7 Optional panning while zooming. In the GUI labeled **EOSView - Swath/Grid Image**, a thumbnail representation of the entire Image will be displayed in the subwindow labeled **Pan Window**. A hollow rectangle on the thumbnail indicates that portion of the Image that is being displayed in the main window. Use the mouse left button to click and drag the rectangle to a new location on the thumbnail image.
 - The portion of the zoomed Image shown in the main window will be the portion indicated by the hollow rectangle on the thumbnail image.
 - The panning will not result in any change to the data in the file.
 - The panning option may be repeated as desired.
 - 8 Optional flatfile output. From the GUI **EOSView—Grid Select**, select **Grid Dimensions** and take note of the PixelsXTrack value (the number of elements in the Grid object's raster); you will need to remember this value in order to make later use of your output flatfile (since the flatfile by definition contains no structural metadata). In the GUI labeled **MyDataField**, click on the **File** menu and then select **Save**. Then select either **Binary** or **ASCII** flatfile type, and assign the output file a name.
 - This procedure will create an output flatfile (which may be quite large, depending on the size of the original Grid object!), but will not result in any change to the data in the original HDF-EOS file.
 - You will later need to use the IDL Tool (q.v.) to view or manipulate the output flatfile.
 - The flatfile output option may be repeated as desired.
 - 9 To end the session with displaying Grid object, in the GUI labeled **EOSView - Swath/Grid**, click on the **File** menu and select **Close**.
 - The **EOSView - Swath/Grid** GUI will disappear.

Table 16.2.2-1. Viewing HDF-EOS Grid Objects - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	select Grid object	double click
2	Data Fields → OK	(No action)
3	Ydim, Xdim → OK	(No action)
4	File → Make image → Continue	(No action)
5	(Optional) Palette → Select → Default Greyscale Antarctica Rainbow World Colors	choose palette
6	(Optional) Zooming → Select → Bilinear Interpolation Nearest Neighbor	choose resampling mode
7	(Optional) Pan Window	click and drag
8	(Optional) Grid Dimensions → note PixelsXTrack value → Flatfile output → Save → Select → Binary ASCII → name output file	note number of elements in Grid object's raster for later use; choose output flatfile type; assign output flatfile name
9	File → Close	(No action)

16.2.3 Viewing HDF-EOS Swath Objects

This procedure describes how to use the EOSView tool to view science data in the HDF-EOS output file that are in HDF-EOS Swath format. These are generally the science data and not browse images). See Section 14.2 for how to select an HDF-EOS Swath object within an HDF-EOS file. An HDF-EOS file is defined as a file produced using the SDP Toolkit metadata tools and having, at a minimum, the ECS core metadata.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools and that at least one object is an HDF-EOS Swath.
2. EOSView has been properly installed and is accessible to the user.
3. The HDF-EOS file has been read into EOSView by the procedures described in Section 16.2.

To view an HDF-EOS Swath object with the EOSView tool, execute the procedure steps that follow:

- 1 In the GUI labeled **EOSView - MyOutputFile.hdf**, double click on a Swath object listed for which data is to be inspected.
 - A GUI labeled **EOSView - Swath Select** will be displayed.
 - Be patient - this GUI may take some time to appear, particularly for large Swath objects.
 - Clicking on the **Dimensions** checkbox and then clicking on the **OK** button displays dimension information about the selected Swath object such as Dimension Names and Sizes.
 - Clicking on the **Geolocation Mappings** checkbox and then clicking on the **OK** button displays geolocation information about the selected Swath object such as the Geolocation Dimensions, Data Dimensions, Offsets, and Increments.
 - Clicking on the **Indexed Mappings** checkbox and then clicking on the **OK** button displays index mapping information about the selected Swath.
 - Clicking on the **Geolocation Fields** checkbox and then clicking on the **OK** button displays geolocation fields information about the selected Swath object.
 - Clicking on the **Attributes** checkbox and then clicking on the **OK** button displays attribute information about the selected Swath object.
- 2 In the GUI labeled **EOSView - Swath Select**, click on the **Data Fields** checkbox and then click on the **OK** button. Then double click on one of the data fields listed.
 - A GUI labeled **EOSView - Swath - SwathObjectName - Start/Stride/Edge** will be displayed where *SwathObjectName* will be replaced by the name of the Swath object selected in step 1.
- 3 In the GUI labeled **EOSView - Swath - SwathObjectName - Start/Stride/Edge**, click on the checkboxes for both **ScanLineTra** and **PixelsXtrac** and then click on the **OK** button.
 - A GUI labeled **MyDataField** will be displayed where *MyDataField* will be replaced by the name of the data field selected in step 2.
 - The data will be displayed in this GUI in the form of a table of values.
- 4 In the GUI labeled **MyDataField**, click on the **File** menu and then select **Make Image**. Optionally adjust the default minimum and maximum data values and then click on the **Continue** button.
 - Adjusting the minimum and maximum data values will affect only the display (contrast) and not the actual data in the file.
 - A GUI labeled **EOSView - Swath/Grid Image** will appear, displaying the data field in image form.
- 5 Optional colorization. In the GUI labeled **EOSView - Swath/Grid Image**, click on the **Palette** menu, then select **Select** and then select one of the palettes listed: **Default**, **Greyscale**, **Antarctica**, **Rainbow**, or **World Colors**.
 - The selection of palette will not result in any change to the data in the file or to the object's default palette.
 - This selection may be repeated until the desired palette is chosen.
- 6 Optional zooming. In the GUI labeled **EOSView - Swath/Grid Image**, click on the **Zooming** menu, then select **Select** and then select one of the resampling methods listed: **Bilinear Interpolation** or **Nearest Neighbor**. Then click on the **Zoom In** or **Zoom Out** buttons to apply the method.

- The selection of resampling method and zoom will not result in any change to the data in the file.
 - The zooming options may be repeated as desired.
- 7 Optional panning while zooming. In the GUI labeled **EOSView - Swath/Grid Image**, a thumbnail representation of the entire Image will be displayed in the subwindow labeled **Pan Window**. A hollow rectangle on the thumbnail indicates that portion of the Image that is being displayed in the main window. Use the mouse left button to click and drag the rectangle to a new location on the thumbnail image.
- The portion of the zoomed Image shown in the main window will be the portion indicated by the hollow rectangle on the thumbnail image.
 - The panning will not result in any change to the data in the file.
 - The panning option may be repeated as desired.
- 8 Optional flatfile output. From the GUI *EOSView—Swath Select*, select **Swath Dimensions** and take note of the PixelsXTrack value (the number of elements in the Swath object's raster); you will need to remember this value in order to make later use of your output flatfile (since the flatfile by definition contains no structural metadata). In the GUI labeled *MyDataField*, click on the **File** menu and then select **Save**. Then select either **Binary** or **ASCII** flatfile type, and assign the output file a name.
- This procedure will create an output flatfile (which may be quite large, depending on the size of the original Swath object!), but will not result in any change to the data in the original HDF-EOS file.
 - You will later need to use the IDL Tool (q.v.) to view or manipulate the output flatfile.
 - The flatfile output option may be repeated as desired.
- 9 To end the session with displaying Swath object, in the GUI labeled **EOSView - Swath/Grid**, click on the **File** menu and select **Close**.
- The **EOSView - Swath/Grid** GUI will disappear.

Table 16.2.3-1. Viewing HDF-EOS Swath Objects - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	select Swath object	double click
2	Data Fields → OK	(No action)
3	ScanLineTra, PixelsXtrac → OK	(No action)
4	File → Make image → Continue	(No action)
5	(Optional) Palette → Select → Default Greyscale Antarctica Rainbow World Colors	choose palette
6	(Optional) Zooming → Select → Bilinear Interpolation Nearest Neighbor	choose resampling mode
7	(Optional) Pan Window	click and drag
8	(Optional) Swath Dimensions → note PixelsXTrack value → Flatfile output → Save → Binary ASCII → name output file	note number of elements in Swath object's raster for later use; choose output flatfile type; assign output flatfile name
9	File → Close	(No action)

16.2.4 Viewing HDF SDS Objects

This procedure describes how to use the EOSView tool to view science data in the HDF-EOS output file that are in HDF SDS (standard HDF science data set) format. These are generally the science data and not browse images). See Section 14.2 for how to select an HDF SDS object within an HDF-EOS file. An HDF-EOS file is defined as a file produced using the SDP Toolkit metadata tools and having, at a minimum, the ECS core metadata.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The output file is HDF-EOS and has been created and populated with metadata using the SDP Toolkit metadata tools and that at least one object is an HDF SDS.
2. EOSView has been properly installed and is accessible to the user.
3. The HDF-EOS file has been read into EOSView by the procedures described in Section 16.2.

To view an HDF SDS object with the EOSView tool, execute the procedure steps that follow:

- 1 In the GUI labeled **EOSView - MyOutputFile.hdf**, double click on a SDS object listed for which data is to be inspected.

- A GUI labeled **EOSView - Multi-Dimension SDS** will be displayed.
 - A number of checkboxes will be displayed, one for each of the dimensions in the selected SDS (there will be at least two, an X and a Y).
 - Be patient - this GUI may take some time to appear, particularly for large SDS objects.
- 2 In the GUI labeled **EOSView - Multi-Dimension SDS**, click on two of the dimension checkboxes and then click on the **Table** button. Then double click on one of the data fields listed.
- A GUI labeled **MySDS** will be displayed where *MySDS* will be replaced by the name of the SDS object selected in step 1.
- 3 In the GUI labeled **MySDS**, click on the **File** menu and then select **Make Image**. Optionally adjust the default minimum and maximum data values and then click on the **Continue** button.
- Adjusting the minimum and maximum data values will affect only the display (contrast) and not the actual data in the file.
 - A GUI labeled **EOSView - Image Display Window - MySDS** will appear, displaying the data field in image form.
- 4 Optional colorization. In the GUI labeled **EOSView - Image Display Window - MySDS**, click on the **Palette** menu, then select **Select** and then select one of the palettes listed: **Default**, **Greyscale**, **Antarctica**, **Rainbow**, or **World Colors**.
- The selection of palette will not result in any change to the data in the file or to the object's default palette.
 - This selection may be repeated until the desired palette is chosen.
- 5 Optional zooming. In the GUI labeled **EOSView - Image Display Window - MySDS**, click on the **Zooming** menu, then select **Select** and then select one of the resampling methods listed: **Bilinear Interpolation** or **Nearest Neighbor**. Then click on the **Zoom In** or **Zoom Out** buttons to apply the method.
- The selection of resampling method and zoom will not result in any change to the data in the file.
 - The zooming options may be repeated as desired.
- 6 Optional panning while zooming. In the GUI labeled **EOSView - Image Display Window - MySDS**, a thumbnail representation of the entire Image will be displayed in the subwindow labeled **Pan Window**. A hollow rectangle on the thumbnail indicates that portion of the Image that is being displayed in the main window. Use the mouse left button to click and drag the rectangle to a new location on the thumbnail image.
- The portion of the zoomed Image shown in the main window will be the portion indicated by the hollow rectangle on the thumbnail image.
 - The panning will not result in any change to the data in the file.
 - The panning option may be repeated as desired.
- 7 To end the session with displaying Swath object, in the GUI labeled **EOSView - Image Display Window - MySDS**, click on the **File** menu and select **Close**.
- The **EOSView - Image Display Window - MySDS** GUI will disappear.

Table 16.2.4-1. Viewing HDF SDS Objects - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	select SDS object	double click
2	dimensions → Table	(No action)
3	File → Make image → Continue	(No action)
4	(Optional) Palette → Select → Default Greyscale Antarctica Rainbow World Colors	choose palette
5	(Optional) Zooming → Select → Bilinear Interpolation Nearest Neighbor	choose resampling mode
6	(Optional) Pan Window	click and drag
7	File → Close	(No action)

16.3 Viewing Product Data with the IDL Tool

This procedure describes how to use the IDL (Interactive Data Language) COTS tool to inspect the data in the output file from a PGE. These procedures are geared toward binary and ASCII formats, but can be extended to other formats supported by IDL including HDF, NetCDF, and JPEG. Consult the IDL references for details on these other formats. See Section 16.2 for how to inspect the science data and metadata in an HDF-EOS file.

The major activities addresses here include creating an image display (Section 16.3.1), saving an image display (Section 16.3.2), creating a plot display (Section 16.3.3), and saving a plot display (Section 16.3.4).

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The output file is binary, ASCII, or one of the other IDL supported data formats.
2. IDL has been properly installed and is accessible to the user.

To view product data with the IDL tool, execute the procedure steps that follow:

1. From the SSIT Manager, click on the **Tools** menu, then choose **Product Examination**, then **IDL**.
 - An xterm (on the AIT Sun) will be displayed within which the IDL command interpreter will be run.
2. Go to procedure depending upon the activity to perform.
 - Proceed to Section 16.3.1 to create an image display.
 - Proceed to Section 16.3.2 to save an image display.
 - Proceed to Section 16.3.3 to create a plot display.

- Proceed to Section 16.3.4 to save a plot display.
- 3 To end the IDL session, close any display windows remaining, then at the IDL prompt type **quit**, press **Return**.
- The IDL session will be closed.

Table 16.3-1. Viewing Product Data with the IDL Tool - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<u>T</u> ools → <u>P</u> roduct Examination → <u>I</u> DL	(No action)
2	(No entry)	choose an IDL activity
3	quit	press Return

16.3.1 Creating an Image Display Using IDL

This procedure describes how to use the IDL Tool to create an image display. The next procedure, Section 16.3.2, describes how to save an image display (once created) to either a data file or a graphic file. For creating and saving plot displays, see Sections 16.3.3 and 16.3.4, respectively.

The Activity Checklist table that follows provides an overview of the process for creating an image display using the IDL Tool. Column one (**Order**) shows the order in which tasks should be accomplished. Column two (**Role**) lists the Role/Manager/Operator responsible for performing the task. Column three (**Task**) provides a brief explanation of the task. Column four (**Section**) provides the Procedure (P) section number or Instruction (I) section number where details for performing the task can be found. Column five (**Complete?**) is used as a checklist to keep track of which task steps have been completed.

Table 16.3.1-1. Creating an Image Display Using IDL - Activity Checklist

Order	Role	Task	Section	Complete?
Binary Data:				
1	SSI&T	Open binary data input file.	(I) 16.3.1	
2	SSI&T	Create an image array for the data.	(I) 16.3.1	
3	SSI&T	Read binary data into image array.	(I) 16.3.1	
4	SSI&T	Display the image.	(I) 16.3.1	
5	SSI&T	Load a color table.	(I) 16.3.1	
6	SSI&T	Close the input file.	(I) 16.3.1	
ASCII Data:				
1	SSI&T	Open ASCII data input file.	(I) 16.3.1	
2	SSI&T	Create an image array for the data.	(I) 16.3.1	
3	SSI&T	Read ASCII data into image array.	(I) 16.3.1	
4	SSI&T	Display the image.	(I) 16.3.1	
5	SSI&T	Load a color table.	(I) 16.3.1	
6	SSI&T	Close the input file.	(I) 16.3.1	
JPEG Data:				
1	SSI&T	Read JPEG formatted file into image array.	(I) 16.3.1	
2	SSI&T	Load a color table.	(I) 16.3.1	
3	SSI&T	Display the image.	(I) 16.3.1	

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. IDL is running (see Section 16.3).
2. The PGE output file to be examined is of an IDL-supported type/format (if in doubt, consult the *IDL Reference Guide*).
3. For binary files, data is assumed to be 8-bit characters. Follow the steps listed under **Binary Data**.
4. For ASCII files, data is assumed to be comma-delimited characters. Follow the steps listed under **ASCII Data**.
5. For JPEG (Joint Photographic Expert Group) formatted files, dimension information is carried in the file header and therefore it does not need to be known. Follow the steps listed under **JPEG Data**.

To create an image display using IDL, execute the procedure steps that follow:

Binary Data:

- 1 At the IDL prompt, type **OPENR,1,('MyBinaryFilename')**, press **Return**.

- The *MyBinaryFilename* is the full path name and file name of the binary data file of known dimensions to read in.
 - The single quotes (') must be included around the path/file name.
 - The **1** is the logical unit number.
- 2 At the IDL prompt, type ***MyImage*=BYTARR(*dimx*,*dimy*)**, press **Return**.
 - The *MyImage* is the name to be given to the image once created.
 - The *dimx* and *dimy* are respectively the X and Y dimensions of the input data.
 - 3 At the IDL prompt, type **READU,1,*MyImage***, press **Return**.
 - 4 At the IDL prompt, type **TV,*MyImage***, press **Return**.
 - The image, *MyImage*, should then be displayed.
 - Alternatively, type **TV,*MyImage*,*offsetx*,*offsety***, where *offsetx* and *offsety* are respectively the element and line offsets from *MyImage*'s origin (because of the way IDL defines an image's origin, you may need to use negative values for these offsets).
 - 5 At the IDL prompt, type **LOADCT,3**, press **Return**.
 - This command loads color table number 3. Other color tables are available; refer to the *IDL Reference Guide* for more details.
 - 6 At the IDL prompt, type **CLOSE,1**, press **Return**.
 - This closes logical unit 1.
 - Always close logical units or an error will result the next time an access is attempted.

ASCII Data:

- 1 At the IDL prompt, type **OPENR,1,('MyASCIIfilename')**, press **Return**.
 - The *MyASCIIfilename* is the full path name and file name of the ASCII data file of known dimensions to read in.
 - The single quotes (') must be included around the path/file name.
 - The **1** is the logical unit number.
- 2 At the IDL prompt, type ***MyImage*=BYTARR(*dimx*,*dimy*)**, press **Return**.
 - The *MyImage* is the name to be given to the image once created.
 - The *dimx* and *dimy* are respectively the X and Y dimensions of the input data.
- 3 At the IDL prompt, type **READF,1,*MyImage***, press **Return**
- 4 At the IDL prompt, type **TV,*MyImage***, press **Return**.
 - The image, *MyImage*, should then be displayed.
 - Alternatively, type **TV,*MyImage*,*offsetx*,*offsety***, where *offsetx* and *offsety* are respectively the element and line offsets from *MyImage*'s origin (because of the way IDL defines an image's origin, you may need to use negative values for these offsets).
- 5 At the IDL prompt, type **LOADCT,3**, press **Return**.

- This command loads color table number 3. Other color tables are available; refer to the *IDL Reference Guide* for more details.
- 6 At the IDL prompt, type **CLOSE,1**, press **Return**.
- This closes logical unit 1.
 - Always close logical units or an error will result the next time an access is attempted.

JPEG Data:

- 1 At the IDL prompt, type **READ_JPEG,"MyJPEGfilename.jpg",MyImage**, press **Return**.
- The *MyJPEGfilename* is the full path name and file name of the JPEG formatted data file.
 - The double quotes (") must be included around the path/file name.
 - The *MyImage* is the name to be given to the image created.
- 2 At the IDL prompt, type **TVLCT,r,g,b**, press **Return**.
- Note that r,g,b color table syntax is used for most formatted file types in IDL.
- 3 At the IDL prompt, type **TV,MyImage**, press **Return**.
- The image, *MyImage*, should then be displayed.

Table 16.3.1-2. Creating an Image Display Using IDL - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
Binary Data:		
1	OPENR,1,('MyBinaryFilename')	press Return
2	MyImage=BYTARR(dimx,dimy)	press Return
3	READU,1,MyImage	press Return
4	TV,MyImage	press Return
5	LOADCT,3	press Return
6	CLOSE,1	press Return
ASCII Data:		
1	OPENR,1,('MyASCIIfilename')	press Return
2	MyImage=BYTARR(dimx,dimy)	press Return
3	READF,1,MyImage	press Return
4	TV,MyImage	press Return
5	LOADCT,3	press Return
6	CLOSE,1	press Return
JPEG Data:		
1	READ_JPEG,"MyJPEGfilename.jpg",MyImage	press Return
2	TVLCT,r,g,b	press Return
3	TV,MyImage	press Return

16.3.2 Saving an Image Display Using IDL

This procedure describes how to use the IDL Tool to save an image display. The previous procedure, Section 16.3.1, describes how to create an image display. For creating and saving plot displays, see Sections 16.3.3 and 16.3.4, respectively.

The Activity Checklist table that follows provides an overview of the process for saving an image display using the IDL Tool. Column one (**Order**) shows the order in which tasks should be accomplished. Column two (**Role**) lists the Role/Manager/Operator responsible for performing the task. Column three (**Task**) provides a brief explanation of the task. Column four (**Section**) provides the Procedure (P) section number or Instruction (I) section number where details for performing the task can be found. Column five (**Complete?**) is used as a checklist to keep track of which task steps have been completed.

Table 16.3.2-1. Saving an Image Display Using IDL - Activity Checklist

Order	Role	Task	Section	Complete?
Binary Data:				
1	SSI&T	Open binary data output file.	(I) 16.3.2	
2	SSI&T	Write the output file.	(I) 16.3.2	
3	SSI&T	Close the output file.	(I) 16.3.2	
ASCII Data:				
1	SSI&T	Open ASCII data output file.	(I) 16.3.2	
2	SSI&T	Write the output file.	(I) 16.3.2	
3	SSI&T	Close the output file.	(I) 16.3.2	
JPEG Data:				
1	SSI&T	Write the output file.	(I) 16.3.2	

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. IDL is running (see Section 16.3).
2. The image display is to be saved in a binary (8-bit) or ASCII (comma-delimited characters) format. For other output formats, consult the *IDL Reference Guide*.
3. For binary file option, data will be output as 8-bit characters. Follow the steps listed under **Binary Data**.
4. For ASCII file option, data will be output as comma-delimited characters. Follow the steps listed under **ASCII Data**.
5. For JPEG (Joint Photographic Expert Group) file option, data will be output as a JPEG-formatted file (lossy-compressed). Follow the steps listed under **JPEG Data**.

To save an image display using IDL, execute the procedure steps that follow:

Binary Data:

- 1 At the IDL prompt, type **OPENW,1,('MyBinaryFilename.bin')**, press **Return**.
 - The *MyBinaryFilename.bin* is the full path name and file name of the binary data file to write out.
 - The single quotes (') must be included around the path/file name.
 - The **1** is the logical unit number.
- 2 At the IDL prompt, type **WRITEU,1,MyImage**, press **Return**.
 - The *MyImage* is the name of the image to save.
- 3 At the IDL prompt, type **CLOSE,1**, press **Return**.
 - This closes logical unit 1.
 - Always close logical units or an error will result the next time an access is attempted.

ASCII Data:

- 1 At the IDL prompt, type **OPENW,1,('MyASCIIfilename.asc')**, press **Return**.
 - The *MyASCIIfilename.asc* is the full path name and file name of the ASCII data file to write out.
 - The single quotes (') must be included around the path/file name.
 - The **1** is the logical unit number.
- 2 At the IDL prompt, type **PRINTF,1,MyImage**, press **Return**.
 - The *MyImage* is the name of the image to save.
- 3 At the IDL prompt, type **CLOSE,1**, press **Return**.
 - This closes logical unit 1.
 - Always close logical units or an error will result the next time an access is attempted.

JPEG Data:

- 1 At the IDL prompt, type **WRITE_JPEG,'MyJPEGfilename.jpg',MyImage**, press **Return**.
 - The *MyJPEGfilename.jpg* is the full path name and file name of the JPEG data file to write out.

Table 16.3.2-2. Saving an Image Display Using IDL - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
Binary Data:		
1	OPENW,1,('MyBinaryFilename.bin')	press Return
2	WRITEU,1,MyImage	press Return
3	CLOSE,1	press Return
ASCII Data:		
1	OPENW,1,('MyASCIIfilename.asc')	press Return
2	PRINTF,1,MyImage	press Return
3	CLOSE,1	press Return
JPEG Data:		
1	WRITE_JPEG,"MyJPEGfilename.jpg",MyImage	press Return

16.3.3 Creating a Plot Display Using IDL

The procedures for creating a plot display are clearly described in the IDL manuals; some exceptions are clarified below.

Setting axis limits for a plot:

- 1 At the IDL prompt, type **SURFACE,MyPlot,AX=70,AZ=70,xrange=[0,20],yrange=[0,20]zrange=[0,30]**, and press **Return**.
 - The *MyPlot* is the IDL session variable (to which you have assigned some math function, program output, image, etc.).
 - *AX* sets the displayed rotation about the X axis.
 - *AZ* sets the displayed rotation about the Z axis.
 - The values of *xrange* set the displayed portion of the X axis.
 - The values of *yrange* set the displayed portion of the Y axis.
 - The values of *zrange* set the displayed portion of the Z axis.
 - The plot will then be displayed to the screen.

Setting axis titles for a plot:

- 1 At the IDL prompt, type **SURFACE,MyPlot,AX=70,AZ=70,xtitle='this is X',ytitle='this is Y',ztitle='this is Z'**, and press **Return**.
 - The *MyPlot* is the IDL session variable (to which you have assigned some math function, program output, image, etc.).
 - The value of *xtitle* sets the displayed title of the X axis.
 - The value of *ytitle* sets the displayed title of the Y axis.
 - The value of *ztitle* sets the displayed title of the Z axis.
 - The plot will then be displayed to the screen.

16.3.4 Saving a Plot Display Using IDL

Saving a displayed plot to a permanent file:

- 1 At the IDL prompt, type **MyPlotDisplay=SURFACE,MyPlot,AX=80,AZ=20**, and press **Return**.
 - The *MyPlotDisplay* is session name for the displayed plot of *MyPlot*.
 - The *MyPlot* is the IDL session variable (to which you have assigned some math function, program output, image, etc.).
- 2 At the IDL prompt, type **SAVE,MyPlotDisplay,4,'MyPlotOutput.ps'**, press **Return**.
 - The *MyPlotDisplay* is the session name of the plot display .
 - The *MyPlotOutput.ps* is the desired name for the saved file.
 - The SAVE option number 4 sets the output file type to PostScript (ps). There are other options, of course (consult the IDL manuals).

16.3.5 Raster Math Fundamentals Using IDL

Most of this subject is covered in the IDL Manuals; some exceptions are described below.

Putting Raster Math results to a temporary display:

The following steps assume that you have already created two IDL session images, say “imageA” and “imageB”, having the same raster size (same number of lines and elements).

- 1 At the IDL prompt, type **imageC=imageA+imageB-14.8**, and press **Return**.
 - The *imageC* is session name for the result of the Raster Math shown above. The operation indicated is performed on each pixel.
 - IDL also supports other Raster Math operators—multiplication (*), division (/), exponent (^).
- 2 At the IDL prompt, type **TV, imageC**, press **Return**.
 - This displays the session image imageC, which can then be saved to a permanent file, etc.
- 3 Alternatively (to steps 1 and 2), at the IDL prompt, type **TV,imageA*7.4/(imageB^8.2+1)**, press **Return**.
 - This shortcut displays the result of the indicated operation, but does not give you the option to either print the result from a session image or to save it as a permanent file. It may be useful (since your IDL session has a limited capacity for holding session images) if you are just experimenting with different Raster Math operations and do not intend to print or save your results.

16.4 Raster Mapping Fundamentals

This procedure describes how to use the IDL Tool to perform basic raster mapping functions. These are two-dimensional spatial functions involving map projections, rather than surface modeling (also called “2.5D”—for which see Plotting section and consult IDL manuals), volumetric modeling (also called “3D”—for which consult the IDL manuals), or two-dimensional spectral functions (for which see Raster Math section and consult the IDL manuals). Note that the pixel-level effects of Raster Mapping functions on an image are typically non-reversible; you can change an image’s map projection from Projection A to Projection B and back again, but you won’t get exactly the same image you started with (hint--you should make a back-up copy of your original image before engaging in Raster Mapping).

The Activity Checklist table that follows provides an overview of the process for raster mapping fundamentals using the IDL Tool. Column one (**Order**) shows the order in which tasks should be accomplished. Column two (**Role**) lists the Role/Manager/Operator responsible for performing the task. Column three (**Task**) provides a brief explanation of the task. Column four (**Section**) provides the Procedure (P) section number or Instruction (I) section number where details for performing the task can be found. Column five (**Complete?**) is used as a checklist to keep track of which task steps have been completed.

Table 16.4-1. Raster Mapping Fundamentals - Activity Checklist

Order	Role	Task	Section	Complete?
Global Data Set Image:				
1	SSI&T	Display the global data set image.	(I) 16.4	
2	SSI&T	Set the map projection.	(I) 16.4	
3	SSI&T	Create a new image using map projection.	(I) 16.4	
4	SSI&T	Display new image.	(I) 16.4	
5	SSI&T	Optionally, overlay Lat/Lon grid.	(I) 16.4	
6	SSI&T	Optionally, overlay world coastlines.	(I) 16.4	
Sub-Global Data Set Image:				
1	SSI&T	Display the sub-global data set image.	(I) 16.4	
2	SSI&T	Set the map projection with limits.	(I) 16.4	
3	SSI&T	Create a new image within limits using map projection.	(I) 16.4	
4	SSI&T	Display new image.	(I) 16.4	
5	SSI&T	Optionally, overlay Lat/Lon grid.	(I) 16.4	
6	SSI&T	Optionally, overlay world coastlines.	(I) 16.4	

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. IDL is running (see Section 16.3).

2. For a global data set image, one having coordinates defined from -180 to 180 degrees East Longitude and 90 to -90 degrees North Latitude, follow the steps listed under **Global Data Set Image**.
3. For a sub-global data set image, one having coordinates defined for subintervals of longitude and latitude (*e.g.* from -88 to -77 degrees East Longitude and 23 to 32 degrees North Latitude), follow the steps listed under **Sub-Global Data Set Image**.

To perform basic raster mapping for the cases listed above using IDL, execute the procedure steps that follow:

Global Data Set Image:

- 1 At the IDL prompt, type **TV,MyImage**, press **Return**.
 - The *MyImage* is the image name of the global image data set.
 - The image, *MyImage*, should then be displayed.
- 2 At the IDL prompt, type **MAP_SET,/ORTHOGRAPHIC**, press **Return**.
 - IDL also supports other map projections. Refer to *IDL Reference Guide*.
- 3 At the IDL prompt, type **MyNewImage=MAP_IMAGE(MyImage,startx,starty,/BILIN)**, press **Return**.
 - The *MyNewImage* is the name to assign to the resulting image.
 - The *MyImage* is the name of the original global image data set.
 - IDL also supports other resampling methods besides Bilinear Interpolation (*/BILIN*). Refer to *IDL Reference Guide*.
- 4 At the IDL prompt, type **TV,MyNewImage,startx,starty**, press **Return**.
 - The image *MyNewImage* should then be displayed.
- 5 Optional overlay Lat/Lon. At the IDL prompt, type **MAP_GRID**, press **Return**.
 - This overlays Lat/Lon graticule onto *MyNewImage*.
- 6 Optional overlay world coastlines. At the IDL prompt, type **MAP_CONTINENTS**, press **Return**.
 - This overlays world coastlines onto *MyNewImage*.
 - Note that the IDL world coastline vector file is itself approximate; the match between this vector coastline and your image's own coastline will therefore also be approximate, but the potential mismatch will usually be too small to notice on a global display.

Sub-Global Data Set Image:

- 1 At the IDL prompt, type **TV,MyImage**, press **Return**.
 - The *MyImage* is the image name of the sub-global image data set.
 - The image, *MyImage*, should then be displayed.

- 2 At the IDL prompt, type **MAP_SET,/MERCATOR,LIMIT=[lat1,lon1,lat2,lon2]**, press **Return**.
 - The *lat1*, *lon1*, *lat2*, and *lon2* specify the latitude and longitude intervals of the sub-global image data set. For example, type **MAP_SET,/MERCATOR,LIMIT=[23,-88,32,-77]**, press **Return**.
- 3 At the IDL prompt, type **MyNewImage=MAP_IMAGE(MyImage,startx,starty,/BILIN,LATMIN=lat1,LATMAX=lat2,LONMIN=lon1,LONMAX=lon2)**, press **Return**.
 - The *MyNewImage* is the name to assign to the resulting image.
 - The *MyImage* is the name of the original global image data set.
 - The *lat1*, *lon1*, *lat2*, and *lon2* specify the latitude and longitude intervals of the sub-global image data set.
 - IDL also supports other resampling methods besides Bilinear Interpolation (*/BILIN*). Refer to *IDL Reference Guide*.
- 4 At the IDL prompt, type **TV,MyNewImage,startx,starty**, press **Return**.
 - The image *MyNewImage* should then be displayed.
- 5 Optional overlay Lat/Lon. At the IDL prompt, type **MAP_GRID**, press **Return**.
 - This overlays Lat/Lon graticule onto *MyNewImage*.
- 6 Optional overlay world coastlines. At the IDL prompt, type **MAP_CONTINENTS**, press **Return**.
 - This overlays world coastlines onto *MyNewImage*.
 - Note that the IDL world coastline vector file is itself approximate; the match between this vector coastline and your image's own coastline will therefore also be approximate, but the potential mismatch will usually be too small to notice on a continental display. If your display is subcontinental, you may observe a noticeable mismatch between the vector and image coastlines—at this level of detail the approximate nature of the IDL vector file becomes noticeable, and the difference (if any) between underlying Earth Model (ellipsoid or horizontal datum) of the vector file and your map-projected image can add a substantial (up to 1+ km) additional mismatch to the display.

Table 16.4-2. Raster Mapping Fundamentals - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
Global Data Set Image:		
1	TV, <i>MyImage</i>	press Return
2	MAP_SET,/ORTHOGRAPHIC	press Return
3	<i>MyNewImage</i> =MAP_IMAGE(<i>MyImage</i> ,startx,starty,/BILI N)	press Return
4	TV, <i>MyNewImage</i> ,startx,starty	press Return
5	(Optional) MAP_GRID	press Return
6	(Optional) MAP_CONTINENTS	press Return
Sub-Global Data Set Image:		
1	TV, <i>MyImage</i>	press Return
2	MAP_SET,/MERCATOR,LIMIT=[<i>lat1,lon1,lat2,lon2</i>]	press Return
3	<i>MyNewImage</i> =MAP_IMAGE(<i>MyImage</i> ,startx,starty,/BILI N.LATMIN= <i>lat1</i> ,LATMAX= <i>lat2</i> ,LONMIN= <i>lon1</i> ,LONMAX= <i>lon2</i>)	press Return
4	TV, <i>MyNewImage</i> ,startx,starty	press Return
5	(Optional) MAP_GRID	press Return
6	(Optional) MAP_CONTINENTS	press Return

This page intentionally left blank.

17. Placing the Science Software Executable on the Data Server

17.1 Assembling a Science Software Executable Package

This section describes how to assemble a Science Software executables Package (SSEP) and create a corresponding Target MCF. A SSEP is a UNIX tar file which contains PGE executables, SDP Toolkit message files, and a Bourne shell profile, if applicable.

A Bourne shell profile file contains any external environment variables needed by the PGE at runtime. The file **must** be named **profile**.

For example:

```
BRAND=sgi

export BRAND

LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/ecs/<mode>/imsl/vni/lib
/lib.sgi:/usr/ecs/<mode>/sybase/lib

export LD_LIBRARY_PATH
```

In order to Insert a SSEP into the Data Server (Section 17.2), a corresponding Target MCF must be generated for the SSEP. Such an ASCII metadata ODL file can be obtained by editing an existing template ODL file with the information of the specific PGE. The following procedures describe how to assemble a SSEP and create an ASCII metadata ODL file.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. PGE executables, message files, and the Bourne shell profile (if necessary) are available to make a SSEP.

To create a SSEP, execute the steps that follow:

- 1 At the UNIX prompt on an AIT Sun, type **mkdir *SSEPpathname***, press **Return**.
 - The *SSEPpathname* is the full path name of a *new* directory which will contain all the files to be placed into the SSEP as well as the SSEP itself.
 - It is recommended that *SSEPpathame* be named with a convention that indicates the PGE for which a SSEP will be created. For example, type **mkdir MOD35.ssep**, press **Return**.
- 2 At the UNIX prompt on the AIT Sun, type **cd *SSEPpathname***, press **Return**.

- The *SSEPpathname* is the directory name of the new directory created in step 1.
- 3 At the UNIX prompt on the AIT Sun, type **cp *pathname/file1 pathname/file2 ... pathname/filen .***, press **Return** (note the “dot” and then space at the end of the command).
- The *pathname/file1,pathname/file2,...pathname/filen* represents a list of path names and file names (delimited by spaces) to copy into the current directory, *SSEPpathame* (the “dot” represents the current directory and must be last in the command).
 - For example, type **cp /data/MODIS/pge/MOD35.pge /data/MODIS/mcf/mod35.mcf /data/MODIS/MOD_13453 .**, press **Return** (note the space and then “dot” at the end of the command).
 - The files copied into this directory should be the PGE executable, any shell scripts or other executables that are part of the PGE, SDP Toolkit message files, and the Bourne shell profile (if applicable).
 - Files can be individually copied into the *SSEPpathame* directory. For example, type **cp /data/MODIS/pge/MOD35.pge .**, press **Return** (note the space and then “dot” at the end of the command). Repeat for each file needed in the SSEP for this PGE.
- 4 At the UNIX prompt on the AIT Sun, type **tar cvf *SSEPfilename.tar* ***, press **Return**.
- The *SSEPfilename.tar* is the file name for the SSEP tar file. The file name extension .tar is recommended but not required.
 - The asterisk (*) is a file name wildcard that represents all files in the current directory. This will place all files in the SSEP tar file.
 - Once created, the contents of the SSEP tar file can be viewed by typing **tar tvf *SSEPfilename.tar***, press **Return**.
 - Do not apply compression (e.g. UNIX compress or gzip) to the tar file.
- 5 At the UNIX prompt on the AIT Sun, type **cp *filename.met.tpl filename.met***, press **Return**.
- The *filename.met.tpl* is the file name of the template Target MCF for this SSEP. If a template is not available, see Appendix D or use one used for another SSEP.
 - The *filename.met* is the file name of the Target MCF to be tailored for this SSEP.
- 6 At the UNIX prompt on the AIT Sun, type **vi *filename.met***, press **Return**.
- The *filename.met* is the Target MCF for this SSEP.
 - This command invokes the vi editor. Edit the *filename.met* with the specific information for the SSEP to be inserted.
 - The following guidelines should be followed when editing on the Target MCF (*filename.met*):
 - The value for the VERSIONID object should be filled out with the proper PGE version. For example: “1”.
 - In the INFORMATIONCONTENTCONTAINER object,
 - The value for the PARAMETERNAME object of the class “1” should be filled out with the PGE name. For example: “BTS”.
 - The value for the PARAMETERNAME object of the class “2” should be filled out with the PGE Science Software Version. For example: “1”.

- The value for the PARAMETERNAME object of the class “3” should be filled out with the Platform Name. For example: “IRIX”.
 - The value for the PARAMETERNAME object of the class “4” should be filled out with the Platform Version. For example: “6.2”.
 - The value for the PARAMETERNAME object of the class “5” should be filled out with the date to perform the Insertion. For example: “970319”.
 - The value for the PARAMETERNAME object of the class “6” should be filled out with the time to perform the Insertion. For example: “14:45:00”.
 -
 - A Target MCF is shown in listing 17.1.2-1.
- 7 Save the changes made to the SSEP’s Target MCF (*filename.met*) and exit the editor.
- The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.
 - For other editors, refer to that editor’s documentation.

Table 17.1-1. Assembling a Science Software Executable Package- Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<code>mkdir SSEPpathname</code>	press Return
2	<code>cd SSEPpathname</code>	press Return
3	<code>cp pathname/file1 pathname/file2 ... pathname/filen .</code>	press Return
4	<code>tar cvf SSEPfilename</code>	press Return
5	<code>cp PGENAME#vers.tpl filename.met</code>	press Return
6	<code>vi filename.met</code>	edit specific values for VALUE= in the filename.met if they are blank.
7	<code>:wq</code>	press Return

Listing 17.1-1. Template Target MCF for a SSEP

```

/*****
/*****
/* This is a template for the metadata file for a PGE Exe Tar */
/* inserted into the Data Server. It must be filled out by the */
/* SSIT operator before the PGE Exe Tar file can be inserted. */
/* The relevant fields that need to be filled out are */
/* called out by comments and the actual information should */
/* replace the <>. */
/* */
/*****
/*****
GROUP = INVENTORYMETADATA
  GROUPTYPE = MASTERGROUP
  GROUP = CollectionDescriptionClass
    OBJECT = ShortName
      NUM_VAL = 1
      Value = "PGEEXE"
    END_OBJECT = ShortName
  OBJECT = VersionID
    NUM_VAL = 1
    Value = "1"
  END_OBJECT = VersionID
END_GROUP = CollectionDescriptionClass

GROUP = AdditionalAttributes
  OBJECT = AdditionalAttributesContainer
    Class = "1"
  OBJECT = AdditionalAttributeName
    NUM_VAL = 1
    Value = "ExePGENAME"
  END_OBJECT = AdditionalAttributeName

  /*****
  /* InformationContent needs to contain the PGE Name */
  /* in the Value field. */
  /*****
  GROUP = InformationContent
    Class = "1"
  OBJECT = ParameterValue
    NUM_VAL = 1
    Value = "Mike's PGE"
  END_OBJECT = ParameterValue
  END_GROUP = InformationContent
END_OBJECT = AdditionalAttributesContainer
OBJECT = AdditionalAttributesContainer
  Class = "2"

```

```

OBJECT = AdditionalAttributeName
  NUM_VAL = 1
  Value = "ExePGESSWVersion"
END_OBJECT = AdditionalAttributeName

/*****
/* InformationContent needs to contain PGE SW Version */
/* in the Value field. */
*****/
GROUP = InformationContent
  Class = "2"
  OBJECT = ParameterValue
    NUM_VAL = 1
    Value = "SW#1"
  END_OBJECT = ParameterValue
END_GROUP = InformationContent
END_OBJECT = AdditionalAttributesContainer
OBJECT = AdditionalAttributesContainer
  Class = "3"
  OBJECT = AdditionalAttributeName
    NUM_VAL = 1
    Value = "ExePlatformOS"
  END_OBJECT = AdditionalAttributeName
/*****
/* InformationContent needs to contain the Platform OS value */
/* in the Value field. */
*****/
GROUP = InformationContent
  Class = "3"
  OBJECT = ParameterValue
    NUM_VAL = 1
    Value = "IRIX6.2"
  END_OBJECT = ParameterValue
END_GROUP = InformationContent
END_OBJECT = AdditionalAttributesContainer
OBJECT = AdditionalAttributesContainer
  Class = "4"
  OBJECT = AdditionalAttributeName
    NUM_VAL = 1
    Value = "ExePlatformOSVersion"
  END_OBJECT = AdditionalAttributeName

/*****
/* InformationContent needs to contain the OS version value */
/* in the Value field. */
*****/
GROUP = InformationContent
  Class = "4"
  OBJECT = ParameterValue

```

```

        NUM_VAL = 1
        Value = "6.2"
        END_OBJECT = ParameterValue
    END_GROUP = InformationContent
END_OBJECT = AdditionalAttributesContainer

OBJECT = AdditionalAttributesContainer
Class = "5"
OBJECT = AdditionalAttributeName
    NUM_VAL = 1
    Value = "ExeInsertDate"
    END_OBJECT = AdditionalAttributeName

/* **** */
/* InformationContent needs to contain Insert Date value */
/* in the Value field. */
/* **** */
GROUP = InformationContent
    Class = "5"
    OBJECT = ParameterValue
        NUM_VAL = 1
        Value = "1997-01-10T12:12:00.000000Z"
        END_OBJECT = ParameterValue
    END_GROUP = InformationContent
    END_OBJECT = AdditionalAttributesContainer
END_GROUP = AdditionalAttributes
END_GROUP = INVENTORYMETADATA
END

```


17.2 Inserting a Science Software Executable Package to the Data Server

Science software, like any other data that are managed in the ECS, must be placed on the Data Server. A program called the SSEP Insertion Tool can be used for Inserting a Science Software Executable Package into the Data Server.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The ESDT descriptor file for the SSEP to be Inserted exists in the ODL format on the Data Server.
2. A Target MCF for this SSEP has been created for the Insert.
3. The SSEP has been created according to the procedures in Section 17.1.

To Insert the SSEP to the Data Server, execute the steps that follow:

- 1 From the SSIT Manager, click on the **T**ools menu, then choose **D**ata **S**erver and then **I**nsert **E**XE **T**AR.
 - An xterm in which DpAtInsertExeTarFile.sh is running will be displayed.
 - Alternatively, the same tool can be invoked by typing at a UNIX prompt on an AIT Sun **DpAtInsertExeTarFile.sh**, press **Return**.
 - The DpAtInsertExeTarFile.sh program will prompt for further information.
- 2 At the program prompt **Configuration filename (default defaultConfigFile)?**, press **Return**.
 - The default configuration file for this tool will be used.
 - The **defaultConfigFile** will be replaced by the full path name and file name of the default configuration file. The file name will be DpAtIE_*daac*.CFG where *daac* will be replaced by one of {GSFC, EDC, LARC, NSIDC}.
- 3 At the program prompt **ECS mode of operation (enter for default: OPS)?**, press **Return**.
 - If **OPS** is not the correct mode, then enter in the correct mode (*e.g.* SSIT) and press **Return**.
- 4 At the program prompt **Name of PGE?**, type **PGENAME**, press **Return**.
 - The **PGENAME** is the name of the PGE for which this static granule is being Inserted. For example, type **PGE01**, press **Return**.
 - The **PGENAME** must match exactly the PGE name entered into the PDPS for this PGE (see Section 13.1.2).
- 5 At the program prompt **Science software version of PGE?**, type **SSWversion**, press **Return**.

- The *SSWversion* is the version of the science software which is being Inserted in this SSEP. Press **Return** to accept the default or enter in a version and press **Return**.
- 6 At the program prompt **Staged filename to insert (including Full path)?**, type *pathname/SSEPFileName*, press **Return**
- The *pathname/SSEPFileName* is the full path name and file name of the SSEP tar file to be Inserted. For example, type */data/MOD35/ssep/MOD35_1.tar*, press **Return**.
 - The SSEP tar file must not be compressed (*e.g.* with UNIX compress or gzip).
- 7 At the program prompt **Associated ASCII metadata filename to insert (including Full Path)?** *pathname/SSEPFileName.met*?, press **Return**.
- The default is the file name of the granule to insert with the .met file name extension. If the default is not correct, then the file name of this file must be entered.
- 8 At the program prompt **Top level shell filename within tar file?**, type *ExecFileName*, press **Return**.
- The *ExecFileName* is the file name of the top level executable within the SSEP tar file. It should be the same as was entered into the PDPS/SSIT Database Update GUI (Section 13.1.3).
 - The SSEP will be Inserted to the Data Server.
- 9 At the program prompt **Hit return to run again, 'q <return>' to quit:** type **q** and press **Return** to quit or just press **Return** to insert additional dynamic granules.
- If continuing, repeat steps 3 through 8.

Table 17.2-1. Inserting a Science Software Executable Package to the Data Server - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	Tools → Data Server → Insert EXE TAR	(No action)
2	<i>Configuration FileName</i>	press Return
3	<i>Mode</i>	press Return
4	<i>PGName</i>	press Return
5	<i>SSWversion</i>	press Return
6	<i>pathname/SSEPFileName</i>	press Return
7	<i>pathname/SSEPFileName.met</i>	press Return
8	<i>ExecFileName</i>	press Return
9	q Return	press Return

18. Inserting a Science Software Archive Package

The Science Software Archive Package (SSAP) is a grouping of science software, documentation, and other related files that is stored at the DAAC. They are accessed and updated through an SSIT SSAP GUI. The SSAP is supposed to be a record of the software, complete with source code, documentation, what and how it was tested, etc., and is created both for recording purposes and so that the tests can/could be repeated later. Note that the executables and any static files needed by the PGE are stored separately from the SSAP.

The SSAP is similar, but not identical to the DAP (Delivery Algorithm Package), in that it contains test data, source code, etc. Much of what is in the DAP will make it into the SSAP (although it may be modified, i.e., code may have bug fixes). Basically, the DAP is what arrives at the SSIT doorstep, while the SSAP is a similarly packaged finished product of the SSIT process, and so contains some things that were not in the DAP (and vice versa).

The SSAP will be made up of 2 different Data Types at the Data Server. The Algorithm Package is metadata about the SSAP, such as the name of the PGE, name of the instrument, date accepted, etc. Each part of the SSAP (source code, documentation, test data) will be stored as an SSAP component, with its own metadata in addition to the files. SSAP components such as source code will be tarred to retain the directory structure (so they must be untarred when retrieved). The executables and static files are stored separately from the SSAP, and thus have their own Data Types (ESDTs).

What follows is a list of items in the SSAP taken from the DID205). See also the Core Metadata model under DAP for a graphical representation of the SSAP.

The following make up an SSAP:

- Documentation
 - Delivery Memo
 - Summary Information for each PGE.
 - System Description Document (SDD).
 - Operations Manual
 - Processing Files Description Document
 - Test Plans (these include the test cases)
 - Scientific documents
 - Interface Definition Document
 - Detailed Performance Testing Results
 - Detailed design/implementation documents
 - COTS User or Programmer Guides
- Software & Control files:
 - Science software source code (including make files & scripts)
 - Testing software source code (including make files & scripts)
 - Test Data Input (this may only be the UR for this)
 - Expected Test Output

- Coefficient Files
- Process Control File
- Metadata Configuration File
- ODL files. These define the PGE and its related Data Types to the PDPS database. They don't currently have official names.
- Other files:
 - A change log created by the SSAP GUI to track changes to the SSAP.

18.1 Inserting an SSAP into PDPS

This procedure describes how to insert an SSAP into PDPS.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. Data Server, Storage Management Servers/Services must be up and operational.
2. The session is on a machine from which the component files needed for the SSAP (see list above) are accessible.
3. The C shell (or a derivative) is the current command shell.

FORCHECK is available only on the AIT Suns.

To create the SSAP, execute the procedure steps that follow:

- 1 If not already on an AIT Sun, log into one from your machine.
- 2 Launch the SSIT Manager. (see Section 7.2).
- 3 From the SSIT Manager choose *Tools* menu and then *Data Server* submenu. Choose *SSAP Editor*.
 - The GUI starts. Note that it will first query Data Server for a list of SSAPs that have previously been created. This list will appear in the window at the center (if any SSAPs already exist). Current SSAP field will be blank, and only Refresh and Create buttons will be active. All three tabs (Main, Files, and Metadata) will be active.
- 4 Click on *Create* to create a new SSAP.
 - The Create SSAP window appears. If no OK button is visible, resize the window so that the OK button is visible.
- 5 Enter the name of the SSAP in the first field.
- 6 Enter SSAP version in the second field. Note that version has a limit of 20 characters.
- 7 Click *OK* and the window disappears.

- On the main GUI, the SSAP created (what was entered in the step above) will appear. Current SSAP is now set to that value. All buttons are now active.
- 8 Click on the *File List tab* to set up SSAP components.
 - The File List Tab displays files in the local directory to the left and files in the selected SSAP component to the right. On the bottom left is a directory listing and a method to move through the directory tree on the local machine. Delete and Reset buttons -- both active -- are to the right.
 - 9 Click on the *File Type* button to select the SSAP component to manipulate.
 - 10 Choose one of the menu items.
 - 11 Select a file (or files) from the left window to add to the component.
 - 12 Click the *Add Arrow* button to add the files. They will appear in the right window because they are now part of that SSAP Component.
 - 13 Now select the *Metadata* tab to set the metadata for the new SSAP. The Metadata Tab displays the metadata for the new SSAP. Only the Name and Version will be filled in automatically. The rest of the fields will have default information. While the SSAP can be submitted with the default information, it is wise to fill in valid values. To change a value, Click the mouse in the field you wish to change and type in a new value. For dates click in the first box or use the up/down arrows to move the date up or down. When finished entering a date, click the *OK* button. For text fields just hit the *Enter* key. The button marked “Edit Assoc Collections” on the bottom of the window must be hit and an Associated Collection entered for the SSAP.
 - 14 Click the *Edit Assoc Collections* button.
 - The Edit Associated Collections window displays a list of associated collections and fields for the entry of new ShortNames and Versions (which make up an Associated Collection).
 - 15 Enter a shortname (of an ESDT that has been installed in the Data Server) — must be eight or fewer characters. Note that the Data Server will verify if the Shortname exists.
 - 16 Enter the version (of the installed ESDT).
 - 17 Click *OK* and the new entry to the collection should appear in the window.
 - 18 Click *Done* to close the window.
 - 19 Click on the Metadata tab.
 - 20 Click *Save* to save the updated metadata.

21 Click *Main tab* to get back to the Main tab.

22 Click *Submit* to send the new SSAP to Data Server. When finished, a message should pop up that says “SSAP Successfully inserted to the Data Server”.

19. Inserting a Static File from Outside the DAP

Some static files needed by a PGE are not included in the DAP. These may belong to another EOS instrument, or come from outside the EOS project entirely.

19.1 Inserting a DAP ancillary file into PDPS

This procedure describes how to insert a DAP ancillary file into PDPS.

Detailed procedures for tasks performed by the SSI&T operator are provided in the text that follows.

Assumptions:

1. Data Server, Storage Management Servers/Services must be up and operational.
2. The ESDT of the static file must have been inserted into the Data Server.
3. A metadata file has been created for the static file you wish to insert. To do this, an MCF (See “Getting an MCF in this section”) must be gotten from the Data Server for the ESDT of the file. Mandatory fields are filled into the MCF, creating a metadata file.

To insert the file, execute the procedure steps that follow:

- 1 If not already on an AIT Sun, log into one from your machine.
- 2 Launch the SSIT Manager. (see Section 7.1).
- 3 From the SSIT Manager choose *Tools* menu and then *Data Server* submenu. Choose *Insert Static File*. [The tool can also be executed by being in `/usr/ecs/<MODE>/CUSTOM/bin/DPS` on the SSIT machine. Source the `buildrc` file for the mode in which you are working (*source .buildrc*--note that this only has to be done once per login). Then execute `DpAtInsertStaticFile.sh .`]

Shell script prompts user for information:

- 4 Enter the location of the `DpAtInsertStaticFile` configuration file (`../cfg/DpAtIS.CFG`).
- 5 Enter the `MODE` of operation (`<MODE>`).
- 6 Enter the short name of the ESDT (for the static file). This value is in the `pdps` database under the `PIDataTypeMaster` table and must be in the PGE ODL file.
- 7 Enter the version of the ESDT for the static file. This value is also in the `pdps` database under the `PIDataTypeMaster` table and must be in the PGE ODL file.

- 8** Enter the science group for this static (this will be from the ODL created during Populating the PGE information in the Database).
- 9** Enter the name of the Static file to insert, including full path.
- 10** Enter the name of metadata file to insert, including full path.

A good status message should a result. Information about the Static File should be entered in the `PIDataTypeGranule` table in the PDPS database (including the UR of the Static File). The Static File can now be acquired by DPS when it is an input to a PGE.

20. Updating a Science Software Archive Package

The Science Software Archive Package (SSAP) is a grouping of science software, documentation, and other related files that is stored at the DAAC. For a discussion of the SSAP and its contents, see Section 18, “SSAP insert.”

20.1 Updating an SSAP

This procedure describes how to update an existing SSAP in PDPS.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. Data Server, Storage Management Servers/Services must be up and operational.
2. An SSAP must already have been inserted into the Data Server.
3. The C shell (or a derivative) is the current command shell.

FORCHECK is available only on the AIT Suns.

To update the SSAP, execute the procedure steps that follow:

- 1 If not already on an AIT Sun, log into one from your machine.
- 2 Launch the SSIT Manager. (see Section 7.1).
- 3 From the SSIT Manager choose *Tools* menu and then *Data Server* submenu. Choose *SSAP Editor*.
 - The GUI starts. Note that it will first query Data Server for a list of SSAPs that have previously been created. This list will appear in the window at the center (if any SSAPs already exist--if not, one will, of course, have to be created before the remainder of this procedure can be performed). Current SSAP field will be blank, and only Refresh and Create buttons will be active. All three tabs (Main, Files, and Metadata) will be active.
 - There is currently missing functionality in the SSAP Editor, so before updating the new SSAP you must hit the *Refresh* button to refresh the data about the new SSAP.
- 4 Click on the *Metadata* tab to update the SSAP.
 - The Metadata Tab displays the metadata for the SSAP. All fields will be set to the values entered when the SSAP was created, and the Algorithm Name field will be

grayed out (because it may not be updated). If you want to create a new SSAP from the an existing one, go back to the Main tab and hit the Create With button.

- 5 Click on the Algorithm Version field (currently called Algorithm Description) and enter a new version (different from what is in the field when the tab is clicked).
- 6 Update any other fields that you wish to change. You can even add a new Associated Collection by clicking on the Assoc Collection button and following the steps described in Creating an SSAP.
- 7 Before you leave the Metadata tab, click *Save* to save the updated metadata.
- 8 Click on the *File List* tab to set up new SSAP components.
 - The File List Tab displays files in the local directory to the left and files in the selected SSAP component to the right. On the bottom left is a directory listing and a method to move through the directory tree on the local machine. Delete and Reset buttons -- both active -- are to the right.
- 9 Click on the *File Type* button to select the SSAP component to manipulate.
- 10 Choose one of the menu items.
- 11 Select a file (or files) from the left window to add to the component.
- 12 Click the *Add Arrow* button to add the files. They will appear in the right window because they are now part of that SSAP Component.
- 13 Click *Main* to get back to the Main tab.
- 14 On the Main tab, click *Submit* to send the new SSAP to Data Server. When finished, a message should pop up that says “SSAP Successfully inserted to the Data Server”. The SSAP has been updated at the Data Server.

21. Post-SSI&T Activities

21.1 Validating Inventory Metadata Updates

21.2 Reporting Problems

This section describes procedures for reporting problems that occur during SSI&T. Both science software and ECS problem reporting are handled with the same COTS tool, PureDDTS (Distributed Defect Tracking System) by PureAtria Software, Inc. At some DAACs, two separate copies of DDTS may be used with one copy configured for reporting ECS problems and the other configured for reporting science software problems. At some DAACs, the same copy of DDTS may be used for both tasks with science software problem reporting set up as a distinct “project” having a unique DDTS configuration. In any case, there will likely be DAAC unique features associated with problem reporting of science software. What follows is therefore a generic description.

The environment set up for DDTS is described in Section 21.2.1. Section 21.2.2 describes how to submit a new problem report. Section 21.2.3 describes how to modify an existing problem report.

21.2.1 Setting Up the Environment for DDTS

This procedure describes how to set up the environment for using DDTS. See Section 21.2.2 for how to use DDTS for submitting science software problem reports.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The user has authorization to use the science software configured DDTS.
2. The C shell or a derivative (*e.g.* T shell) is the current user shell.

To set up the environment for using DDTS for science software reporting, execute the procedure steps that follow:

- 1 At a UNIX prompt on an AIT Sun, type **vi \$HOME/.cshrc**, press **Return**.
 - This command invokes the *vi* editor and reads in the *.cshrc* file from the user's home directory.
 - Any text editor may be used such as *emacs*. For example, **emacs \$HOME/.cshrc**, press **Return**.
- 2 In the editor add the following lines if not already there:

```
#DDTS NCR Tool path
if (“`echo $path | grep /home/ddts/bin`” == “”) then
    set path=($path /home/ddts/bin)
endif
```

- The first line, beginning with the hash mark (#) is a comment line.
 - Note the back quote marks (`) inside the double quotes (“”).
 - The above lines should be entered into the .cshrc file, one per line as shown.
- 3 Save the changes made to the .cshrc file and exit the editor.
 - The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.
 - For other editors, refer to that editor’s documentation.
 - 4 At the UNIX prompt on the AIT Sun, type **source \$HOME/.cshrc**, press **Return**.
 - This command causes the .cshrc file to get run and the new commands in it to be executed.
 - Optionally, the user may logout and then log back in. The result will be the same as above.

Table 21.2.1-1. Setting Up the Environment for DDTS - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	vi \$HOME/.cshrc	press Return
2	#DDTS NCR Tool path if (“`echo \$path grep /home/ddts/bin`” == “”) then set path=(\$path /home/ddts/bin) endif	add lines to .cshrc file
3	Save and quit the editor	invoke editor command
4	source \$HOME/.cshrc	press Return

21.2.2 Submitting a New Problem Report

This procedure describes how to submit either a science software or an ECS problem report using DDTS. See Section 21.2.1 for how to set up environment for DDTS.

- 1 From the SSIT Manager, click on the **Tools** menu, then choose **Xterm**. Then telnet to the DDTS server.
 - Alternatively, in any currently available xterm window, spawn a new session: type **xterm &**, press **Return**. Then telnet to the DDTS server.
 - The DDTS server should be known, if not, seek SA.
- 2 At the UNIX prompt on the DDTS server, type **setenv DISPLAY hostname:0.0**, press **Return**.
 - The *hostname* is the name of the machine on which the DDTS GUI is to be displayed, *i.e.* the machine that your are using.
 - If the machine your are on is the DDTS server, this step should not be necessary; the variable should be set by default to **:0.0**, meaning the machine that you are on.

- To verify a setting, type **echo \$DISPLAY**, press **Return**.
- 3 At the UNIX prompt on the DDTS server, type **xddts**, press **Return**.
- The DDTS GUI should be displayed with three distinct windows labeled: **PureDDTS 3.2.1, Record, and Enclosure**.
 - If all three windows do not appear, then a default set up does not exist. In this case, execute the following steps (otherwise, go on to step 4):
 1. Click on the **Select** menu and then choose **Select Project, Class & State....**
A GUI labeled **Select Project, Class & State** will be displayed.
 2. Under the label **Which Class**, click on the **Change Class** button and then choose **Release_A**. Either double click on this entry or single click and then click on the **OK** button.
 3. If the **Change Class** button does not appear, click on the bottom of the scroll bar towards the right to reveal it.
 4. Under the label **Which Projects**, using the scroll bar, click on a project relevant to your problem, for example **PDPS_A**.
 5. Under the **State** field, click on the entry **New** to highlight.
 6. Click on the **Save as Default** button (there will be no visible response).
 7. Finally, click on the **OK** button.
 8. The above parameters set will be saved as your default.
- 4 When entering a new problem report, click on the **File** menu and select **Submit New Record**.
- The GUI labeled **Record** (the middle panel by default) will be refreshed and prompt for input.
- 5 In the **Record** GUI, enter the prompted information.
- At the prompt **Submit to which class of Projects:**, your default project should be listed. If so, just press **Return**. If not, enter the class and then press **Return**.
 - At the prompt **Project name:**, enter a project name and press **Return**. To see list of valid project names, press **Return**, then make a selection in the displayed GUI.
 - At the prompt **NCR Title:**, enter a descriptive title of the problem, then press **Return**. Try to make the problem clear from the title. The title may be free text up to 72 characters in length.
 - At the prompt **Software:**, enter the name of the software or program involved in the problem being reported, then press **Return**. This may be any string up to 20 characters in length.
 - At the prompt **Build Name**, press **Return** to see a list of valid choices. Highlight a selection and click on the **OK** button.
 - At the prompt **Site ID**, press **Return** to see a list of valid choices. Highlight a selection and click on the **OK** button.
 - At the prompt **Test Case ID**, enter an identification of the test case involved in the problem, then press **Return**. This may be any string.
 - At the prompt **Detection Method**, press **Return** to see a list of valid choices. Highlight a selection and click on the **OK** button.
 - At the prompt **Detected-In-Phase**, press **Return** to see a list of valid choices. Highlight a selection and click on the **OK** button. The choices may vary depending on Class.

- At the prompt **Problem Severity (1-5)**, enter the severity of the problem being reported, then press **Return**. A severity of 1 is catastrophic; a severity of 5 is considered minor. Enter **?** and press **Return** to see a list of definitions for each severity level.
 - At the prompt **Do you want to be notified of changes to this defect**, enter **Y** for yes or **N** for no. If yes, you will receive electronic mail notifications when the status of this problem report has been changed.
 - After the above information has been entered, press **Return**. A GUI labeled **Problem** will be displayed.
- 6 In the GUI labeled **Problem**, enter a description of the problem.
- Enough information should be provided to allow developers to isolate the problem and the description of the problem should be clear enough for SSI&T personnel to understand the nature of the problem.
 - The editor provided for entering a problem description is very simple and allows files to be imported and attached to the description.
 - Files such as screen dumps may be placed in the directory `/home/ddts/bin/save-files/` on the DDTS server. The file name of the file can then be included as a Problem Description enclosure.
- 7 When the problem description has been completed, click on the **File** menu and select **Save Changes & Dismiss Editor**.
- Enclosures appear as icons in the enclosures window and can be displayed by double clicking on them.
 - The submitter's UNIX userid and electronic mail address are automatically saved with the problem report.

Table 21.2.2-1. Submitting a New Problem Report - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<code>Tools → Xterm</code>	telnet to DDTS server
2	<code>setenv DISPLAY hostname:0.0</code>	press Return
3	<code>xddts</code>	press Return
4	<code>File → Submit New Record</code>	(No action)
5	<i>class</i> <i>project name</i> <i>NCR title</i> <i>software</i> <i>build name</i> <i>site ID</i> <i>test case ID</i> <i>detection method</i> <i>detected-in-phase</i> <i>problem severity</i> Y N	press Return press Return press Return press Return press Return press Return press Return press Return press Return press Return
6	text description of problem	(No action)
7	<code>File → Save Changes & Dismiss Editor</code>	(No action)

21.2.3 Changing or Revising a Problem Report

Sometimes you may wish to modify or revise an NCR you have previously submitted

- 1 If you are already in the project and problem report skip this procedure. After starting as described in 1-3 above, From the **Select** menu select **Class, Project, & State ...**
- 2 In the window select the NCR you wish to modify. Make the appropriate modifications to the input screen or the enclosure fields.
- 3 Select Submit & Dismiss

This page intentionally left blank.

22. Troubleshooting and General Investigation

An important part of SSI&T is verifying that the output files produced at the DAAC are identical (within particular tolerances) to the test output files delivered with the DAPs. A successful comparison is a strong indication that the porting of the science software from the development facility at the SCF to the operational facility at the DAAC has not introduced any errors.

A number of file comparison tools are available during SSI&T via the SSIT Manager GUI or they can be invoked from the UNIX command line. Two tools are available for comparing HDF or HDF-EOS files, one tool for comparing ASCII files, and another tool for assisting in comparing binary files.

It is assumed that the Instrument Team has delivered test output files (produced at their SCF) with which to perform the comparison.

22.1 Examining PGE Log Files

Three log files are produced by PGEs during runtime: the Status log, User Log, and the Report log. These log files are written by the SDP Toolkit and by the science software using the Toolkit's Status Message Facility (SMF). The location of these log files is specified in the Process Control File (PCF). When the PGE is built and run with the SCF version of the SDP Toolkit, the location and file names of the log files can be set as desired. When the PGE is built with the DAAC version of the SDP Toolkit and run within the PDPS, the location and file names of the log files is set by the system in the instantiated PCF.

The Status log file captures all error and status information. The User log file captures a subset of messages which are more informational. The Report log file captures arbitrary message strings sent by the PGE.

Section 22.1.1 describes how to examine log files produced by PGEs that have been built with the SCF version of the SDP Toolkit and run from the command line.

Sections 22.1.2 describes how to examine log files (within the Production History) produced by PGEs that have been built with the DAAC version of the SDP Toolkit and run within the PDPS.

22.1.1 Log Files From PGEs Run Outside of the PDPS

When the PGE is run outside of the PDPS, the PCF specifies the location and file names of the log files produced. This procedure describes how to locate that information from the PCF and use it to examine the log files.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The PGE has been successfully built with the SCF version of the SDP Toolkit (Section 11.4).
2. The PGE's PCF has been updated properly for the DAAC environment (Section 11.1).

To examine log files from a PGE run outside of the PDPS, execute the procedure steps that follow:

- 1** At the UNIX prompt on an AIT Sun or on the SPR SGI, type **cd *PCFpathname***, press **Return**.
 - The *PCFpathname* is the full path name to the location of the PCF used by the PGE for which log files are to be examined.
- 2** At the UNIX prompt on the AIT Sun or on the SPR SGI, type **vi *PCFfilename***, press **Return**.
 - The *PCFfilename* is the file name of the PCF used by the PGE for which log files are to be examined.
 - This brings up the file named *PCFfilename* in the *vi* editor.
 - Any text editor may be used such as *emacs*. For example, **emacs MOD35.pcf**, press **Return**.
- 3** In the editor, search for logical IDs (beginning in the first column) **10100**, **10101**, and **10102**. These are the PCF entries for the LogStatus, LogReport, and LogUser respectively. For each, note the file names in field 2 and the path names in field 3. Then quit the editor.
 - If field 3 is blank, then the location is given by the default location specified in a line above the entries beginning with the “!” character.
- 4** At the UNIX prompt on the SPR SGI, type **vi *StatusLogPathname/filename***, press **Return**.
 - The *StatusLogPathname/filename* is the full path name and file name of the Status log file noted in step 3 associated with PCF logical ID 10100. When finished, quit the editor.
 - Note any error or warning messages in file.
 - Any text editor may be used such as *emacs*. For example, **emacs /PGE/MOD35/LogStatus**, press **Return**.
- 5** At the UNIX prompt on the SPR SGI, type **vi *UserLogPathname/filename***, press **Return**.
 - The *UserLogPathname/filename* is the full path name and file name of the Status log file noted in step 3 associated with PCF logical ID 10101. When finished, quit the editor.
 - Note any error or warning messages in file.
 - Any text editor may be used such as *emacs*. For example, **emacs /PGE/MOD35/LogUser**, press **Return**.

- 6 At the UNIX prompt on the SPR SGI, type **vi *ReportLogPathname/filename***, press **Return**.
- The ***ReportLogPathname/filename*** is the full path name and file name of the Status log file noted in step 3 associated with PCF logical ID 10102. When finished, quit the editor.
 - Note any anomalous messages in file.
 - Any text editor may be used such as *emacs*. For example, **/PGE/MOD35/LogReport**, press **Return**.

Table 22.1.1-1. Log Files From PGEs Run Outside of the PDPS - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	<i>cd PCFpathname</i>	press Return
2	<i>vi PCFfilename</i>	press Return
3	(No entry)	note path and file names for IDs 10100, 10101, and 10102
4	<i>vi StatusLogPathname/filename</i>	press Return, review file, then quit editor
5	<i>vi UserLogPathname/filename</i>	press Return, review file, then quit editor
6	<i>vi ReportLogPathname/filename</i>	press Return, review file, then quit editor

22.1.2 Production History Log Files From PGEs Run Within the PDPS

The Production History (PH) is created during PGE execution within the PDPS and then Inserted into the Data Server upon PGE completion. The PH is a UNIX tar file that includes the PGE log files.

See Section 12.2 for listing of other files contained in the PH.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The PDPS archive configuration area is properly set up.
2. The environment variable **DataSource** contains the full path name to the archive. This is typically **/imf/archive/** or **/imf_data/archive/** and varies at each DAAC.

To examine the Production History PGE log files, execute the procedure steps that follow:

- 1 At the UNIX prompt on the SPR SGI, type **cd \$DataSource/PH**, press **Return**.
 - The **\$DataSource** is an environment variable containing the full path name of the Data Server archive and **PH** is a subdirectory under **\$DataSource** containing the Production History tar files.
 - For example, type **cd \$DataSource/PH**, press **Return**.
- 2 At the UNIX prompt on the AIT Sun or on the SPR SGI, type **ls -al**, press **Return**.

- A list of the current contents will be displayed. These will be Production History tar files.
 - The file names of PH files are named *PGEname#versionMMDDYYhhmm_runtime.tar_UR* where *PGEname* is replaced by the PGE name, *version* is replaced by the PGE version, *MMDDYY* is replaced by the Insert date in the month-day-year format, *hhmm* is replaced by the Insert time in the hours-minutes format, and *UR* is replaced by the Universal Reference. For example, the PH file name for version 2.1 of a SAGE III PGE named sage_1t Inserted on December 14, 1999 at 2:00pm could be:
sage_1t#2.11214991400_runtime.tar_YAAa005Li_19991214135815.
 - Look for the PH of interest.
- 3** At a UNIX prompt on the AIT Sun or on the SPR SGI, type **cp *PHtarFilename* *WorkingPathname***, press **Return**.
- The *PHtarFilename* is the file name of the Production History tar file.
 - The *WorkingPathname* is the full path name to some working directory in which the Production History tar file is to be placed and examined.
- 4** At the UNIX prompt on the AIT Sun or on the SPR SGI, type **cd *WorkingPathname***, press **Return**.
- The *WorkingPathname* is the full path name to the working directory specified in step 3.
- 5** At the UNIX prompt on the AIT Sun or on the SPR SGI, type **tar xvf *PHtarFilename***, press **Return**.
- The *PHtarFilename* is the file name of the Production History tar file in the working directory.
 - This command will untar the Production History tar file, extracting its component files into the current directory.
- 6** At the UNIX prompt on the AIT Sun or on the SPR SGI, type **vi *StatusLogFilename***, press **Return**.
- The *StatusLogFilename* is the file name of the Status log file within the PH. When finished, quit the editor.
 - Note any error or warning messages in file.
 - Any text editor may be used such as *emacs*. For example, **emacs LogStatus**, press **Return**.
- 7** At the UNIX prompt on the AIT Sun or on the SPR SGI, type **vi *UserLogFilename***, press **Return**.
- The *UserLogFilename* is the file name of the User log file within the PH. When finished, quit the editor.
 - Note any error or warning messages in file.
 - Any text editor may be used such as *emacs*. For example, **emacs LogUser**, press **Return**.
- 8** At the UNIX prompt on the AIT Sun or on the SPR SGI, type **vi *ReportLogFilename***, press **Return**.

- The ***ReportLogFilename*** is the file name of the Report log file within the PH. When finished, quit the editor.
- Note any error or warning messages in file.
- Any text editor may be used such as *emacs*. For example, **emacs LogReport**, press **Return**.

Table 22.1.2-1. Production History Log Files From PGEs Run Within the PDPS - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	cd <i>PHArchivePathname</i>	press Return
2	ls -al	press Return
3	cp <i>PHtarFilename WorkingPathname</i>	press Return
4	cd <i>WorkingPathname</i>	press Return
5	tar xvf <i>PhtarFilename</i>	press Return
6	vi <i>StatusLogFilename</i>	press Return, review file, then quit editor
7	vi <i>UserLogFilename</i>	press Return, review file, then quit editor
8	vi <i>ReportLogFilename</i>	press Return, review file, then quit editor

22.1.3 History Log Files From Failed PGEs Run Within the PDPS

The History Log(HL) is created during PGE execution within the PDPS and then Inserted into the Data Server upon failure of the PGE. The HL is a UNIX file that includes the PGE log files, the PCF and PH.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The PDPS archive configuration area is properly set up.
2. The environment variable *DataServer* contains the full path name to the archive.

To examine the History Log files, execute the procedure steps that follow:

- 1 At the UNIX prompt on the SPR SGI, type **cd \$DataServer/FAILPGE**, press **Return**.
 - The **\$DataServer** is an environment variable containing the full path name of the IMF Data Server archive and **FAILPGE** is a subdirectory under **\$DataServer** containing the History Log files.
 - For example, type **cd \$DataServer/FAILPGE**, press **Return**.
- 2 At the UNIX prompt on the SPR SGI, type **ls -al**, press **Return**.
 - A list of the current contents will be displayed. These will be History Log files.
 - The file names of HL files are named *PGEname#versionMMDDYYhhmm_runtime.tar_UR* where *PGEname* is replaced by the PGE name, *version* is replaced by the PGE version, *MMDDYY* is replaced by the Insert date in the month-day-year format, *hhmm* is replaced by the Insert

time in the hours-minutes format, and *UR* is replaced by the Universal Reference. For example, the HL file name for version 2.1 of a SAGE III PGE named sage_1t Inserted on December 14, 1999 at 2:00pm could be:
sage_1t#2.11214991400_runtime.tar_YAAa005Li_19991214135815.

- Look for the HL of interest.
- 3 At a UNIX prompt on the SPR SGI, type **cp *HLFilename WorkingPathname***, press **Return**.
 - The *HLFilename* is the file name of the History Log file.
 - The *WorkingPathname* is the full path name to some working directory in which the History Log file is to be placed and examined.
 - 4 At the UNIX prompt on the SPR SGI, type **cd *WorkingPathname***, press **Return**.
 - The *WorkingPathname* is the full path name to the working directory specified in step 3.
 - 5 At the UNIX prompt on the SPR SGI, type **vi *HLFilename***, press **Return**.
 - The *HLFilename* is the file name of the History Log. When finished, quit the editor.
 - Note any error or warning messages in file.
 - Any text editor may be used such as *emacs*. For example, **emacs *HLFilename***, press **Return**.

Table 22.1.3-1. History Log File From Failed PGEs Run Within the PDPS - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	cd <i>HLArchivePathname</i>	press Return
2	ls -al	press Return
3	cp <i>HLFilename WorkingPathname</i>	press Return
4	cd <i>WorkingPathname</i>	press Return
5	vi <i>HLFilename</i>	press Return, review file, then quit editor

22.2 The Production History

The Production History (PH) is a UNIX tar file that is produced and archived for every run of a PGE in the PDPS. Each PH can be uniquely retrieved from the Data Server.

PH files are located in the \$DataServer/PH directory and have the following naming convention:

PGEname#versionMMDDYYhhmm_runtime.tar_UR

where *PGEname* is replaced by the PGE name, *version* is replaced by the PGE version, *MMDDYY* is replaced by the Insert date in the month-day-year format, *hhmm* is replaced by the Insert time in the hours-minutes format, and *UR* is replaced by the Universal Reference.

For example, a PH file name for version 2.1 of a SAGE III PGE named sage_1t Inserted on December 14, 1999 at 2:00pm would be:

sage_1t#2.11214991400_runtime.tar_YAAa005Li_19991214135815.

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The PDPS archive configuration area is properly set up.
2. The environment variable `DataServer` contains the full path name to the archive.

To examine the Production History file, execute the procedure steps that follow:

- 1 At the UNIX prompt on an AIT Sun or on the SPR SGI, type **cd \$DataServer/PH**, press **Return**.
 - The **\$DataServer** is an environment variable containing the full path name of the IMF Data Server archive and **PH** is a subdirectory under **\$DataServer** containing the Production History tar files.
 - For example, type **cd \$DataServer/PH**, press **Return**.
- 2 At the UNIX prompt on the AIT Sun or on the SPR SGI, type **ls -al**, press **Return**.
 - A list of the current contents will be displayed. These will be Production History tar files.
 - The file names of PH files are named *PGEname#versionMMDDYYhhmm_runtime.tar_UR* where *PGEname* is replaced by the PGE name, *version* is replaced by the PGE version, *MMDDYY* is replaced by the Insert date in the month-day-year format, *hhmm* is replaced by the Insert time in the hours-minutes format, and *UR* is replaced by the Universal Reference. For example, the PH file name for version 2.1 of a SAGE III PGE named sage_1t Inserted on December 14, 1999 at 2:00pm could be:
sage_1t#2.11214991400_runtime.tar_YAAa005Li_19991214135815.
 - Look for the PH of interest.
- 3 At a UNIX prompt on the AIT Sun or on the SPR SGI, type **cp PHtarFilename WorkingPathname**, press **Return**.
 - The **PHtarFilename** is the file name of the Production History tar file.
 - The **WorkingPathname** is the full path name to some working directory in which the Production History tar file is to be placed and examined.
- 4 At the UNIX prompt on the AIT Sun or on the SPR SGI, type **cd WorkingPathname**, press **Return**.
 - The **WorkingPathname** is the full path name to the working directory specified in step 3.
- 5 At the UNIX prompt on the AIT Sun or on the SPR SGI, type **tar xvf PHtarFilename**, press **Return**.
 - The **PHtarFilename** is the file name of the Production History tar file in the working directory.
 - This command will untar the Production History tar file, extracting its component files into the current directory.

- 6 At the UNIX prompt on the AIT Sun or on the SPR SGI, type **vi *PHcomponentFile***, press **Return**.
- The ***PHcomponentFile*** is the file name of one of the PH component files. The component files are:
 - *PGEname#versionMMDDYYhhmm.Log* - Contains the DPR ID, the actual command used to run the PGE, resource usage information, the PGE exit status, and files used by the PGE.
 - *PGEname#versionMMDDYYhhmm.Pcf* - The actual instantiated PCF used when running the PGE.
 - *PGEname#versionMMDDYYhhmm.ProdLog* - Contains the DPR ID, the PGE ID, and resource usage information (same as in the .Log file).
 - *PGEname#versionMMDDYYhhmm.Profile* - Contains the environment variables defined during the execution of the PGE including the contents of the PATH environment variable.
 - *PGEname#versionMMDDYYhhmm.TkReport* - The Report log file, same as is produced when run outside of the PDPS.
 - *PGEname#versionMMDDYYhhmm.TkStatus* - The Status log file, same as is produced when run outside of the PDPS.
 - *PGEname#versionMMDDYYhhmm.TkUser* - The User log file, same as is produced when run outside of the PDPS.
 - *ESDTmmdyyHHMM.met* - All target MCFs for all Inserts on behalf of the PGE. The *ESDT* is the ESDT ShortName into which the file was Inserted, *mmdyy* is the month, day, and year of the Insert and *HHMM* is the time of the Insert.

Table 22.2-1. The Production History - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	cd <i>PHArchivePathname</i>	press Return
2	ls -al	press Return
3	cp <i>PHtarFilename WorkingPathname</i>	press Return
4	cd <i>WorkingPathname</i>	press Return
5	tar xvf <i>PhtarFilename</i>	press Return
6	vi <i>PhcomponentFile</i>	press Return, review file, then quit editor

22.3 Examining PDPS-Related Scripts and Message Files

This section describes how users may access files, in addition to the PGE-produced log files, which are created during the execution of a DPR job and which may hold information useful in tracing processing problems.

Some of these files are written by default to directory paths that can only be accessed on either the SGI processor machine or one of the Sun workstations. More detailed descriptions of these

files and the conditions under which they are generated will be supplied in future Green Book versions.

22.3.1 Examining AutoSys JIL Scripts

JILxxxxxxxx is the Job Information Language (JIL) script that defines the DPR job to **AutoSys** and which must be submitted to the **AutoSys** Database before a DPR job can be run. The name of the file created is system-generated and begins with the characters 'JIL' followed by nine characters (e.g. JILAAaA0066c).

Sample file content:

```
insert_job: 5251_823122483_1
job_type: command
command: /usr/ecs/Rel_A/CUSTOM/data/bin/sgi/DpAtExecutionMain
5251_823122483_1
machine: sprlsgigsfc
std_out_file: /home/cboettch/mockpge_msfc/out/dpat_std.out
std_err_file: /home/cboettch/mockpge_msfc/out/dpat_std.err
profile: /usr/ecs/Rel_A/CUSTOM/data/bin/sgi/DpAtRunProfile.sh
```

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

To examine JILxxxxxxxx scripts on the AIT Sun, execute the procedure steps that follow:

- 1 At the UNIX prompt on an AIT Sun, type **cd JILscriptPathname**, press **Return**.
 - The **JILscriptPathname** is the full path name to the location of the JILxxxxxxxx scripts to be examined.
- 2 At the UNIX prompt on the AIT Sun, type **vi JILscriptFilename**, press **Return**.
 - The **JILscriptFilename** is the file name of the JILxxxxxxxx script to be examined.
 - This brings up the file named **JILscriptFilename** in the *vi* editor.
 - Any text editor may be used such as *emacs*. For example, **emacs JILscriptFilename**, press **Return**.

Table 22.3.1-1. Examining AutoSys JIL Scripts - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	cd JILscriptPathname	press Return
2	vi JILscriptFilename	press Return

22.3.2 Examining Application Log Files

Most of the custom code used during SSI&T routinely produce log files. For example, the SSIT Manager produces a log file named DpAtMgr.log and the tool used to Insert SSEPs to the Data Server (DpAtInsertExeTarFile.sh) produces a log file named DpAtInsertExeTarFile.log. These files are placed in the directory in which the tool was executed. If the SSIT Manager is run from

the user's home directory, then the log files for each of the associated tools will be found in the user's home directory. Log files are produced at the first invocation of the tools, even if no messages are written to them. During subsequent use of the tools, the associated log files will be appended.

Log files are generally named according to the convention:

ApplicationName.log

where *ApplicationName* is replaced with the name of the tool's executable binary. For tools that are shell scripts (*e.g.* .sh files), the shell name is left out of the log file name. For example, the tool `DpAtInsertStaticFile.sh` produces a log file named `DpAtInsertStaticFile.log` and not `DpAtInsertStaticFile.sh.log`.

23. Miscellaneous

23.1 Setting Up Environment for Direct PDPS Database Access

The PDPS database contains many tables containing information about science software running in the production system. The population of some of this information is part of standard SSI&T procedures (for example, see Section 12.2) and no special environment set up is required for these procedures. It may be necessary, however, to gain direct access to the PDPS database from time to time and this procedure describes how to set up the required environment. Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The user has been given read access (at a minimum) for the PDPS database.
2. The C shell or a derivative (*e.g.* T shell) is the current user shell.

To set up an environment for PDPS database access, execute the procedure steps that follow:

- 1 At a UNIX prompt on an AIT Sun or on the SPR SGI, type **vi \$HOME/.cshrc**, press **Return**
 - This brings up the file named `.cshrc` in the *vi* editor.
 - Any text editor may be used such as *emacs*. For example, **emacs \$HOME/.cshrc**, press **Return**.
- 2 In the editor add the following lines if not already there:


```
setenv SYBASE /vendor/sybase
setenv SYBASE /vendor/sybase
setenv SYBROOT $SYBASE/sybooks
setenv EBTRC $SYBROOT/sun5m/ebtrc
setenv DSQUERY computer-server
set path = ($path $SYBASE $SYBASE/bin $SYBROOT/sun5m/bin)
```

 - The *computer-server* is the name of the server at the local DAAC and should be known. If not known, ask your SA or DBA.
 - The above lines should be entered into the `.cshrc` file, one per line as shown.
- 3 Save the changes made to the `.cshrc` file and exit the editor.
 - The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.
 - For other editors, refer to that editor's documentation.
- 4 At the UNIX prompt on the AIT Sun, type **source \$HOME/.cshrc**, press **Return**.
 - This will reinitialize the setting in the `.cshrc` file.

- The environment set up for access to the PDPS database should now be complete.

Table 23.1-1. Setting Up Environment for Direct PDPS Database Access - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	vi \$HOME/.cshrc	press Return
2	setenv SYBASE /vendor/sybase setenv SYBASE /vendor/sybase setenv SYBROOT \$SYBASE/sybooks setenv EBTRC \$SYBROOT/sun5m/.ebtrc setenv DSQUERY <i>computer-server</i> set path = (\$path \$SYBASE \$SYBASE/bin \$SYBROOT/sun5m/bin)	add lines to .cshrc file
3	Save and quit the editor	invoke editor command
4	source \$HOME/.cshrc	press Return

Detailed procedures for tasks performed by the SSI&T operator are provided in the sections that follow.

Assumptions:

1. The particular application makes use of and produces log files.
2. The location of the application log file is known.
3. The name of the application producing the log file is known.

To examine the application log files, execute the procedure steps that follow:

- 1 At the UNIX prompt on an AIT Sun or on the SPR SGI, type **cd *LogFilesPathname***, press **Return**.
 - The ***LogFilesPathname*** is the full path name to the location of a particular application log file. This is typically the directory from which the SSIT Manager or other tool is run.
 - For example, type **cd \$HOME/ceres**, press **Return**.
- 2 At a UNIX prompt on an AIT Sun or on the SPR SGI, type **vi *ApplicationName.log***, press **Return**
 - This ***ApplicationName.log*** is the file name of the log file to examine.
 - Any text editor may be used such as *emacs*. For example, **emacs QAMonitor.log**, press **Return**.
- 3 When finished, exit the editor.
 - The specifics depend upon which editor is being used. If using *vi*, the command sequence to enter is **:wq**, press **Return**.
 - For other editors, refer to that editor's documentation.

Table 23.1-2. Examining Application Log Files - Quick-Step Procedures

Step	What to Enter or Select	Action to Take
1	cd <i>LogFilesPathname</i>	press Return
2	vi <i>ApplicationName.log</i>	press Return
3	:wq	press Return

This page intentionally left blank.